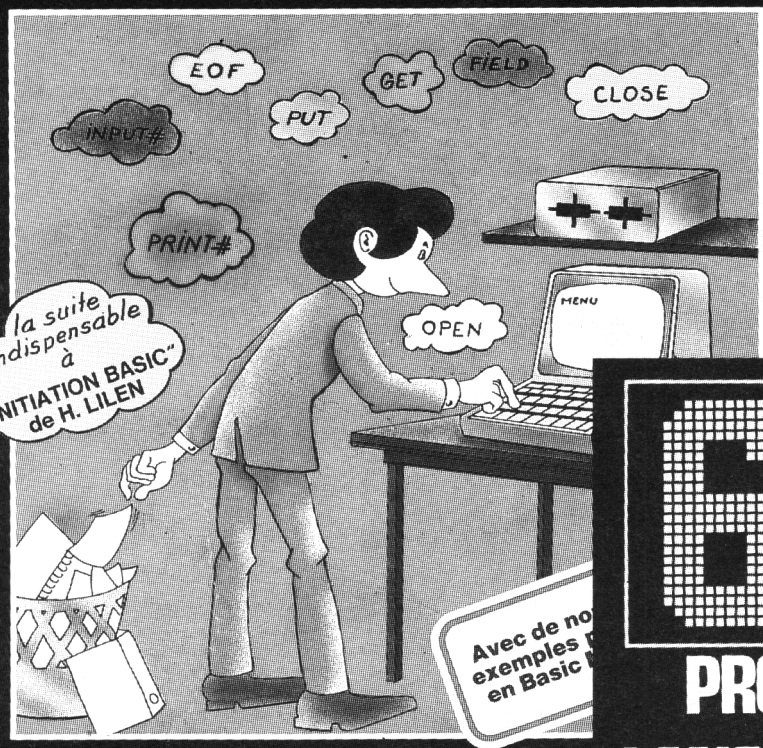
H. LILEN - J. BÉNARD

PRATIQUE de l'ORIC-1 et 36 PROGRAMMES

S. E. C. F.
ÉDITIONS RADIO

INITIATION aux FICHIERS BASIC

J. BENARD



Avec de no
exemples
en Basic

S. E. C. F.

ÉDITIONS RADIO

6502 PROGRAMMATION en LANGAGE ASSEMBLEUR

LANCE A. LEVENTHAL

OSBORNE/McGRAW-HILL

S. E. C. F.

EDITIONS RADIO

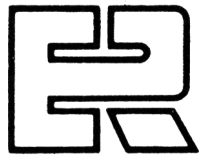
S. E. C. F.

Éditions Radio

H. LILEN - J. BÉNARD

PRATIQUE de l'ORIC-1 et 36 PROGRAMMES

S. E. C. F.



Editions Radio

9, RUE JACOB - 75006 PARIS

TEL. 329.63.70

ATTENTION !

Une erreur peut en cacher une autre...

Le message d'erreur signalé par votre micro-ordinateur pour la ligne 250, par exemple, d'un programme ne signifie pas forcément que cette ligne est mauvaise.

C'est peut-être la ligne 120 qu'il faut corriger, ou tout simplement il y a incompatibilité entre vos données et votre programme.

La loi du 11 mars 1957 n'autorisant, aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective » et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1er de l'article 40). Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code pénal.

© SECF Éditions Radio, Paris
1983

Tous droits de traduction, de reproduction et d'adaptation réservés pour tous pays

Imprimerie Berger-Levrault, Nancy

Dépôt légal : décembre 1983
Éditeur n° 932 - Imprimeur : 778229
I.S.B.N. 2 7091 0932 8

AVANT-PROPOS

Décidément, les Anglais nous surprendront toujours ! Ce sont eux qui, en effet, ont inventé ces merveilleux petits micro-ordinateurs dont l'Oric est le type le plus élaboré, peut-être, et qui connaissent un succès amplement mérité. Grâce à leur faible prix, ces machines ont révélé ce qu'était la micro-informatique à un très vaste public. Des centaines de milliers de personnes, aujourd'hui, les emploient aussi bien pour leurs applications professionnelles ou familiales que pour jouer ou apprendre. Pourquoi pas vous ?
Telle est probablement ce à quoi vous songez si vous avez commencé à feuilleter ce livre : il vous est destiné si vous êtes totalement profane en informatique, si vous n'avez jamais manipulé un micro-ordinateur, si vous ignorez tout de la programmation et du Basic et si même, de surcroît, vous redoutez ces étranges petites machines dont on dit qu'elles sont capables de suppléer l'homme dans ses fonctions intellectuelles.
Nous allons vous prouver que vous pouvez mettre le micro-ordinateur à votre service bien plus aisément que vous ne l'imaginez. Pour cela, nous allons vous proposer :

• *De faire la connaissance de l'Oric-1, pour commencer : comment le brancher, le mettre en service, comment aussi entamer le « dialogue » avec lui.*

• *Puis, nous vous montrerons aussitôt comment utiliser des cassettes de programmes (jeux, enseignement, gestion...) avec un banal magnétophone. Dès lors, et sans même qu'il vous soit besoin d'aller plus avant, vous serez à même d'exploiter* immédiatement *ce micro-ordinateur (sans donc avoir à apprendre la programmation).*

• *Mais nous vous engageons vivement à vous initier à cet art, dans un premier but de démystification. Vous constaterez que la programmation devient très vite l'un des « jeux » les plus passionnants qui soit. Que les informaticiens professionnels nous pardonnent pour ce mot de* jeux *! Pour notre excuse, ajoutons que les ambitions de ce livre sont plus modestes et qu'il n'entend pas, en quelques pages, faire du lecteur un analyste-programmeur de métier. Nous allons toutefois très rapidement vous mettre en demeure de concevoir vos propres programmes dans ce langage appelé le « Basic », qui est, aujourd'hui, le plus répandu.*
Pour cela, laissez-vous simplement guider par les chapitres qui suivent. Toutes les notions nouvelles vont vous être présentées à l'aide d'exemples concrets et d'exercices (donnés avec les résultats que vous devez obtenir), pour que vous les assimiliez sans peine et sans effort inutile de mémoire. Laissez-vous aussi guider par votre fantaisie et votre imagination et n'hésitez pas à créer des variantes des programmes que nous vous proposerons, ce que vous serez certainement tenté de faire. Les auteurs ne peuvent que vous y encourager.

Une question importante se pose toutefois : vous faut-il obligatoirement disposer d'un Oric pour aborder l'étude de ce livre ? Il est évident que c'est préférable car alors, vous pourrez directement expérimenter sur machine et acquérir l'indispensable pratique de l'informatique. Mais ce n'est nullement une condition impérative pour une première lecture rapide, *celle qui doit vous permettre de vous tester vous-même et de vous rassurer pleinement ; elle vous incitera très certainement à franchir le pas et à investir dans l'achat d'une telle machine. Ce dont vous aurez tout lieu de vous féliciter par la suite.*
Les chapitres IV à VIII présentent chacun une partie du langage avec de nombreux exemples brefs qui mettent l'accent sur ce qui est nouveau. A la fin de ces chapitres, vous trouverez des programmes qui mettent en application ces nouveautés. Leur but est de vous donner des idées ; amusez-vous à les essayer, puis modifiez-les.
Enfin, le dernier chapitre regroupe les programmes de récapitulations : c'est une espèce de révision qui vous est proposée ! Selon votre tempérament, vous adopterez sans doute l'une de ces deux démarches : commencer par lire les explications et taper les exemples simples pour tous les chapitres, avant de passer aux programmes d'applications, ou bien tout lire d'une seule traite, en faisant quand même une pause casse-croûte de temps en temps !

CHAPITRE I

POUR FAIRE CONNAISSANCE AVEC L'ORIC

Ce premier chapitre est destiné à vous présenter l'Oric. Plus généralement d'ailleurs, on va vous exposer ce qu'est un micro-ordinateur personnel et de quoi se compose un « système informatique ». On vous montrera aussi comment connecter l'Oric et comment entamer (tout juste !) le dialogue avec lui.

Principaux thèmes étudiés

La micro-informatique recourt à un vocabulaire qui lui est propre. Souvent à base d'anglais, d'ailleurs. Démystifier l'informatique revient souvent à démystifier ce vocabulaire, pour commencer. Nous allons nous y employer dans ce livre en vous présentant progressivement tous les termes qu'il importe de connaître.
Ne les apprenez pas par cœur : si vous hésitez sur la signification d'un mot, reportez-vous au paragraphe qui le définit. Pour cela, nous avons placé, en marge du texte, les termes que nous appellerons des « mots-clés ». Au début de chaque chapitre, nous vous en donnerons la liste dans l'ordre où ils apparaissent. En fin d'ouvrage, tous ces mots-clés sont regroupés par ordre alphabétique, avec l'indication de la page où vous les trouverez cités.
Voici la liste des mots-clés pour ce premier chapitre ; ne vous laissez pas impressionner par une longueur qui n'est qu'apparente !

Branchement de l'Oric – 1	Basic	RAM
Alimentation	Basic Microsoft	Mémoires mortes
Prise Péritel	Chiffre zéro et lettre O	ROM
Mise en service	Microprocesseur	Volatilité des mémoires
QWERTY	Mémoires internes	Ligne physique
AZERTY	Mémoires centrales	Ligne logique
Curseur	Mémoires externes	CTRL
Effacement arrière	Mémoires périphériques	Minuscules
DEL	Cellules mémoires	Majuscules
SYNTAX ERROR	Octet	CTRL/T
Répétition des touches	Caractère	CAPS
SHIFT	Byte	Format de l'écran
Langages	Mémoires vives	RETURN

1. L'Oric – 1 et ses branchements

BRANCHEMENT DE L'ORIC – 1

Oui, le micro-ordinateur Oric – 1, c'est ce clavier que représente notre photo. Ce bloc, mesurant environ 28 cm sur 18 cm, a la forme d'un pupitre de 2 cm d'épaisseur sur l'avant et 5 sur l'arrière ; il contient tous les circuits électroniques qui font un ordinateur.

Afin de le faire fonctionner, il faut l'alimenter en énergie électrique, d'une part, le connecter à un récepteur de télévision pour obtenir des images, d'autre part.

Gros plan sur l'Oric – 1 : son clavier

ALIMENTATION

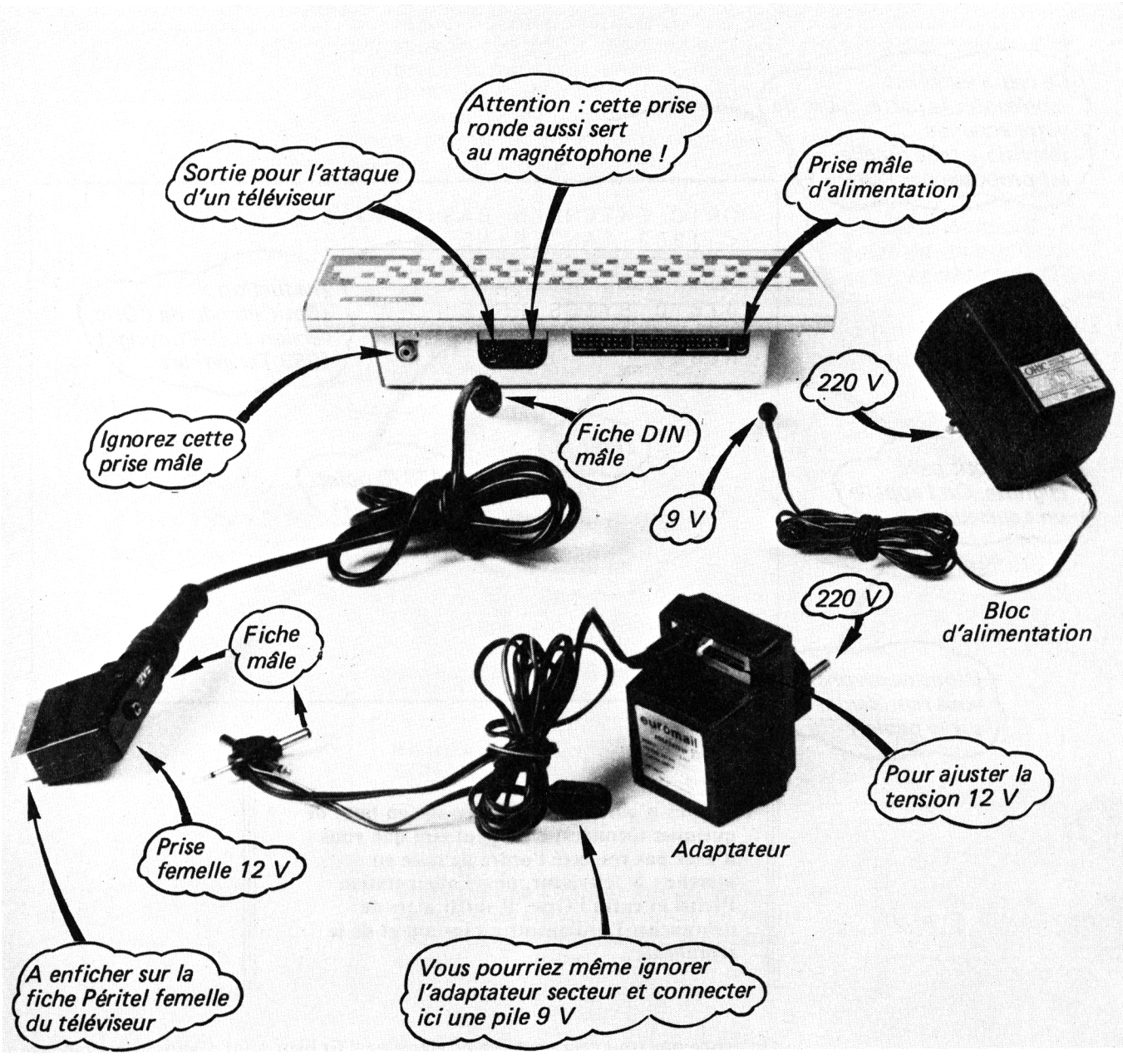
Ce micro-ordinateur a besoin d'une tension continue de 9 volts ; celle-ci lui sera fournie par un petit bloc d'alimentation qu'on connectera directement sur le secteur (220 volts alternatifs) ; il en sort un câble terminé par un connecteur coaxial femelle : insérez-le dans la prise mâle située à l'arrière-droit du micro-ordinateur.

PRISE PERITEL

La connexion à un téléviseur n'est guère plus complexe ; utilisez, de préférence, un récepteur couleur si vous voulez bénéficier de toutes les facultés de votre machine. Le branchement s'effectue via la prise « Péritel » (mot provenant de *péri-télévision*), cette prise étant obligatoire en France sur tous les récepteurs, depuis 1981. C'est une prise à deux rangées de connexions, femelles sur le téléviseur, de forme allongée. Saisissez-vous du câble spécial qui vous est livré avec l'Oric et qui se termine par une fiche mâle correspondante ; enfichez celle-ci. L'autre extrémité de ce câble se termine par une prise ronde, mâle, à 5 broches (dite « aux normes DIN »). Insérez-la dans le connecteur femelle correspondant, situé à l'arrière *gauche* du micro-ordinateur.

Votre micro-ordinateur va ainsi fournir les signaux d'image (rouge, vert et bleu) et de synchronisation à votre téléviseur, mais celui-ci doit basculer en mode *réception vidéo*. Pour cela, il faudra encore amener une tension de 12 volts à la fiche Péritel, à l'aide d'un second bloc d'alimentation, plus petit et ajustable. Il s'agit d'un « adaptateur », à connecter également au secteur 200 V (prévoyez une prise multiple !) ; ajustez l'ergot, qui glisse latéralement sur 12 V (généralement, 6 V suffisent ; vous n'avez rien à redouter si vous essayez diverses valeurs : votre téléviseur ne craint rien car elles seront de toutes façons inférieures à 12 V. Si elles sont trop faibles, toutefois, la commutation interne ne se fera pas et vous n'obtiendrez pas d'image de l'Oric – 1). L'autre extrémité du câble de l'adaptateur se termine par une prise en croix, dont l'intérêt est d'offrir quatre fiches de connexion. Choisissez celle qui s'insère normalement sur la prise femelle située à l'arrière du connecteur mâle Péritel : c'est la plus fine.

La prise mâle située à gauche, à l'arrière du micro-ordinateur, sert à attaquer la fiche antenne d'un récepteur de télévision *aux normes* « PAL ». (Les normes françaises sont appelées SECAM). Ignorez-la donc, et ce d'autant plus allègrement que le branchement Péritel fournit une image de bien meilleure qualité. A moins que vous ne disposiez d'un récepteur de télévision « multistandard », acceptant le PAL et ne possédant pas de prise Péritel, ou encore d'un récepteur SECAM sans prise



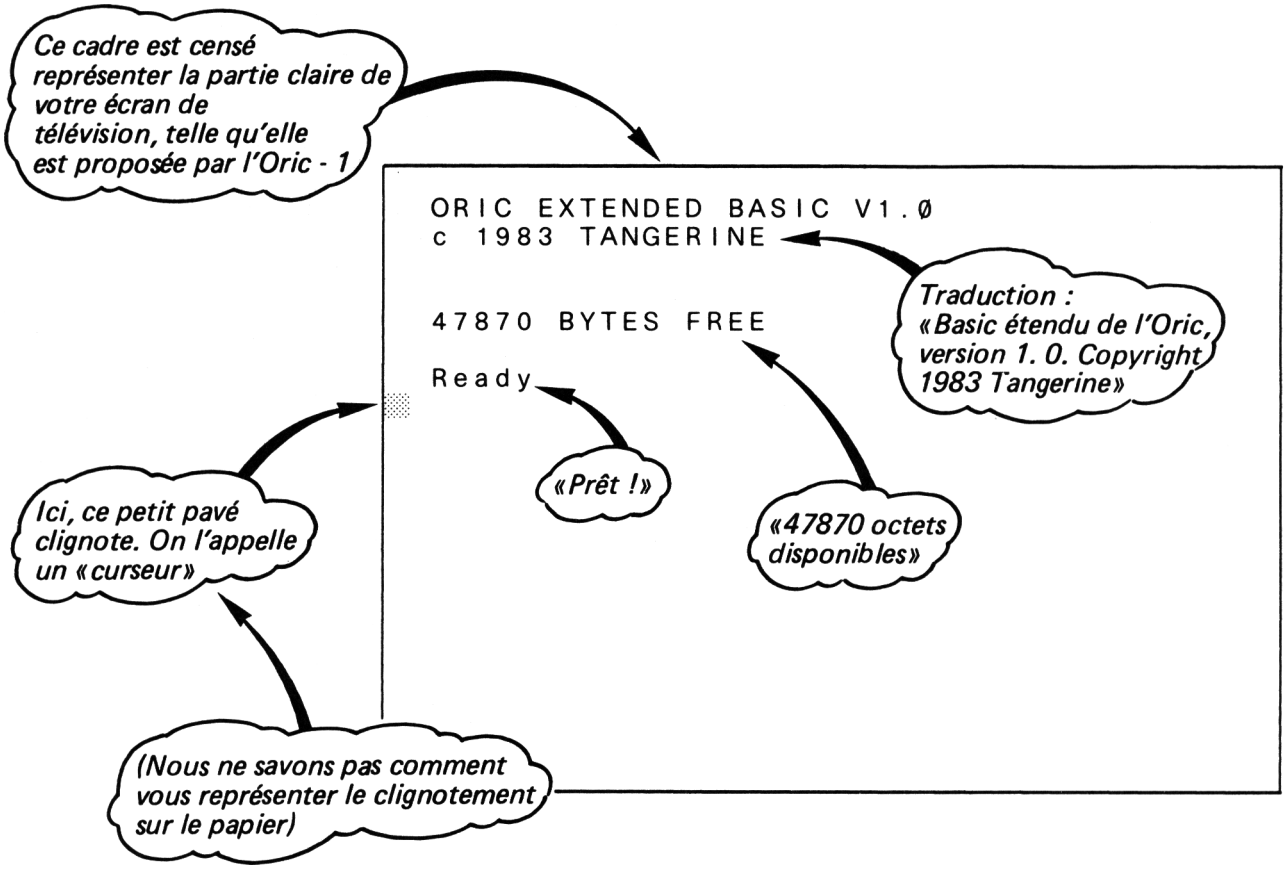
Comment brancher l'Oric – 1 pour le mettre en service.

Péritel : dans ce cas, il vous faudra un boîtier de conversion en plus, ce que votre vendeur sera certainement très heureux de vous fournir (consultez-le).
Cela étant fait, vous voici enfin prêt à faire naître l'image.

2. Mise en service

MISE EN SERVICE

Notez, tout d'abord, que l'Oric – 1 ne possède pas d'interrupteur de mise en service : vous débrancherez simplement ses blocs d'alimentation et adaptateur pour le déconnecter.
Mettez donc le téléviseur en marche, sur n'importe quelle station (625 lignes), peu importe : la commutation Péritel a priorité. Si, à ce moment, vous alimentez l'Oric – 1, vous allez voir apparaître sur l'écran une image quelconque pendant un court instant, puis le message suivant en noir sur fond blanc :



Si vous n'obtenez pas ce message au bout de quelques secondes, c'est peut-être que vous n'avez pas respecté l'ordre de mise en marche : le téléviseur, puis l'alimentation Péritel et enfin l'Oric. Il suffit alors de débrancher l'ordinateur un instant et de le rebrancher.

Qu'est-ce que tout cela peut-il bien signifier ? Et bien, tout d'abord, le « langage » employé par l'Oric se présente poliment : « *Je suis la version 1.Ø du Basic étendu de l'Oric* ». Puis, le micro-ordinateur vous annonce de quelle « capacité mémoire » vous disposez. Enfin, il s'affirme prêt à vous servir. Si le message que vous lisez est différent, ne vous en inquiétez pas : on va voir ce que tout cela signifie.

Réglez donc le contraste et la luminosité du téléviseur afin d'obtenir une image agréable à regarder, puis examinez le clavier. Il comporte cinq rangées de touches marquées de divers symboles.

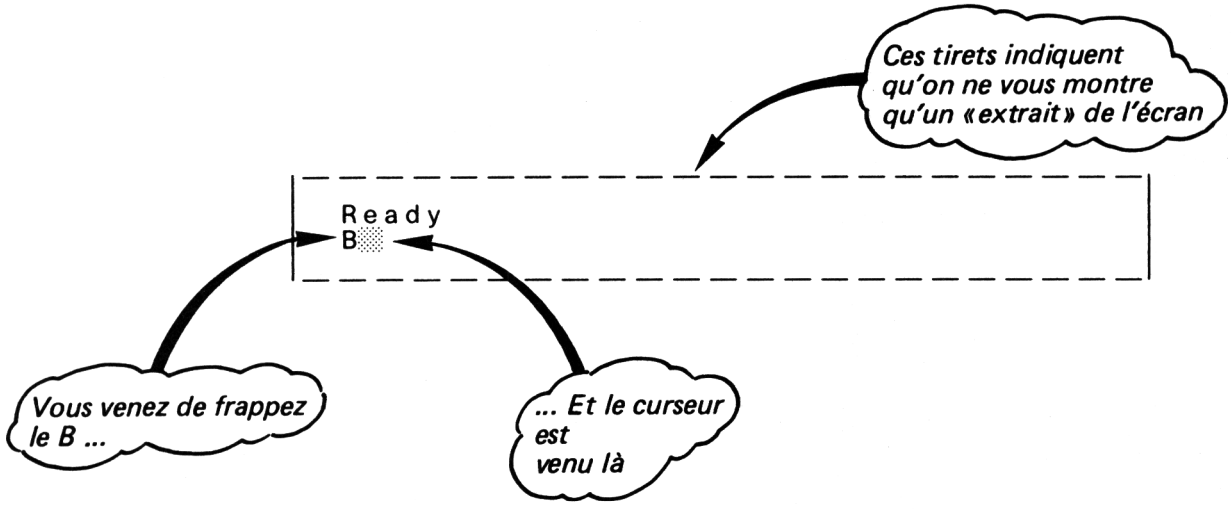
Considérez d'abord la rangée supérieure des touches *alphabétiques*. De gauche à droite, elle commence par les lettres Q, W, E, R, T, Y, etc. ; si, maintenant, vous comparez cette distribution à celle d'une machine à écrire courante, vous verrez qu'à cet emplacement, la machine à écrire dispose des touches A, Z, E, R, T, Y, etc.

QWERTY
AZERTY

La distribution appelée AZERTY correspond aux normes françaises, la distribution QWERTY, aux normes anglaises ou américaines.

CURSEUR

En fait, les différences sont minimes mais capables de gêner une dactylo experte (ou *un* expert du clavier). Si vous êtes aussi peu adroit au clavier que l'auteur, vous n'aurez aucun mal à vous faire au QWERTY. Afin de vous exercer, frappez donc le mot BONJOUR (qui n'a, ici, rigoureusement rien à voir avec l'informatique pure) en examinant ce qui se passe sur l'écran. Tout d'abord, le B se place sur la ligne marquée par le curseur, en 3e position, puis le curseur passe aussitôt à la position suivante :



Frappez donc BONJOUR au complet, ce qui donne :

```
Ready
BONJOUR
```

Remarquez qu'à chaque frappe, le curseur se déplace d'un cran vers la droite : il marque l'emplacement où apparaitra le caractère suivant.

EFFACEMENT ARRIERE

ERROR

Si vous vous trompez en frappant un caractère, appuyez sur la touche marquée DEL (à droite du clavier) ; elle ramène le curseur d'un pas en arrière et ce curseur efface le caractère qu'il recouvre. Vous pouvez dès lors en frapper un nouveau.

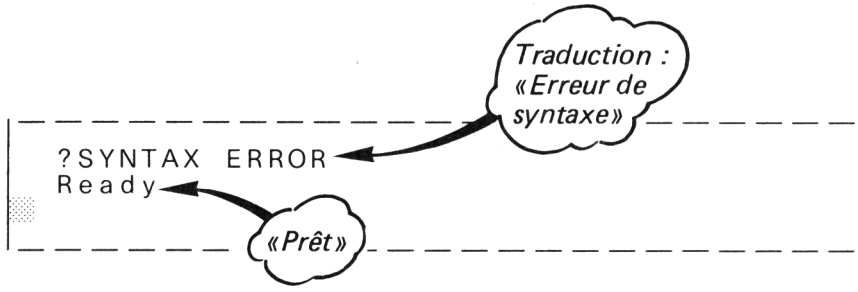
DEL est une touche d'espacement arrière avec effacement.

Comme sur toutes les machines à écrire, le clavier de l'Oric dispose d'une barre d'espacement ; c'est la longue barre horizontale du bas. Elle sert à créer des « blancs », des espaces. Expérimentez-là.

Ne vous privez pas de frapper sur le clavier, même en mélangeant ou combinant des touches. Votre Oric ne craint rien (si vous frappez avec une force « raisonnable ») ; vous ne provoquerez pas de panne !

Cependant, il réagira parfois et affichera des messages bizarres, du genre :

SYNTAX ERROR



C'est tout simplement parce que l'ordinateur n'a pas compris où vous vouliez en venir. Forcément, vous ne respectez pas ses commandes normales (qu'on va vous enseigner plus loin).

A nouveau, ces messages ne nuisent en aucun cas à l'Oric et vous ne pouvez le mettre en panne ainsi.

REPETITION DES TOUCHES

Remarquez que, si vous maintenez une touche enfoncée, son caractère va apparaître une fois mais, après un court instant, il va se répéter en permanence : les touches sont *à répétition.*
Vous avez aussi remarqué que certaines touches sont marquées d'un double symbole, par exemple un chiffre tel que le 1 et, à côté, un point d'exclamation. Si vous frappez sur cette touche (le 1 : en haut et à gauche), vous obtenez un 1 sur l'écran. Mais comme sur une machine à écrire, vous pouvez obtenir le second caractère en « basculant » la frappe. Pour cela, repérez les deux touches marquées SHIFT, de part et d'autre du clavier ; utilisez l'une ou l'autre, selon votre confort :

SHIFT

- Frappez sur SHIFT *et maintenez-la enfoncée.*
- Puis, frappez sur le 1.
- Relâchez le tout.

Vous écrivez ainsi un point d'exclamation (!).

On peut traduire SHIFT par « basculer ». L'action conjointe sur SHIFT et sur une autre touche, le 1 ici, sera écrite par la suite SHIFT/1.

Mais vous en savez assez sur cette mise en route, pour ce premier chapitre. Sachez que nous reviendrons sur le clavier afin de vous en détailler toutes les possibilités.

3. Qu'est-ce que le Basic ?

Un ordinateur se distingue des autres types de machines par le fait qu'on peut lui déléguer des fonctions « intellectuelles ». il sait faire énormément de choses si on les lui « explique » correctement: poser une fusée sur la lune, établir une facture, jouer aux échecs ou vous apprendre les mathématiques... Pour cela, vous devez lui fournir des *instructions* précises développant ce qu'il doit faire dans un *langage* qu'il soit capable de comprendre.

LANGAGES

BASIC

De nombreux « langages » ont été créés, mais le plus répandu en micro-informatique est celui qu'on appelle le *Basic,* qu'on va étudier dans ce livre. Il dérive non pas du français courant (hélas !), mais de l'anglais, car toute l'informatique nous vient d'outre-Atlantique. Rassurez-vous : même si vous n'êtes pas porté sur les langues étrangères, le nombre de mots à acquérir est tellement faible que vous ne vous apercevrez même pas que vous vous exprimez en anglais. Plus que de mots, d'ailleurs, il s'agira de *codes* dont l'apprentissage ne demande pas d'effort.

Le langage Basic a été développé aux Etats-Unis au cours des années 60. Ce mot est formé par les premières lettres de l'expression « *Beginners All-purpose Symbolic Instruction Code* », soit, à peu près : *code d'instructions symboliques tous usages pour débutants.*

Depuis son invention, le Basic a cependant fait souche et s'est multiplié. Aujourd'hui, il existe en de très nombreuses variantes.

BASIC MICROSOFT

Pensez au français, tel qu'il est parlé en France, et au français de Montréal (Canada), par exemple : le noyau de la langue est le même mais ici et là se manifestent des différences sensibles. Il en va de même du Basic, qui existe en plusieurs *dialectes* pourrait-on dire. Le plus répandu des Basics est cependant celui qui a été mis au point par la société américaine *Microsoft ;* c'est celui qu'Oric a adopté. C'est un Basic très complet, *étendu* comme l'on dit, dont la version employée par l'auteur est la 1.Ø, ainsi que l'indiquait l'écran.

CHIFFRE 0 ET LETTRE O

En passant, notez l'écriture « 1.Ø » qui se lit « un point zéro ».
En informatique, on écrira souvent le zéro avec une barre au travers pour le différencier de la lettre O. En effet, pour l'ordinateur, zéro et O sont deux choses totalement différentes. Vérifiez-le en les frappant au clavier. Et ne les confondez pas à l'avenir.

Vous pourrez peut-être disposer d'une version différente du Basic sur votre propre machine, mais peu importe pour ce qui suit. Mais où ce « langage » se trouve-t-il *dans* l'ordinateur ?

4. Microprocesseur et mémoires

Ce qui suit, ici, n'est pas *absolument* indispensable pour utiliser votre micro-ordinateur. Il s'agit d'un petit peu de technique. Mais pourquoi ne pas le lire ? Vous vous ferez, ainsi, des idées plus précises sur les caractéristiques de la machine.

MICROPROCESSEUR

Sachez, tout d'abord, que l'âme de tout micro-ordinateur est un minuscule circuit électronique qui tient dans le creux de la main et s'appelle un « microprocesseur ». Celui qui équipe l'Oric – 1 porte le nom (peu harmonieux) de « 6502 ». C'est lui qui exécute tous vos ordres.

Cependant, il ne peut fonctionner que pour autant qu'il dispose de circuits lui servant de *mémoires.* Dans ces mémoires seront rangés les programmes, les données, les chiffres à additionner ou à soustraire, par exemple, et les résultats. Il s'agit, là encore, de minuscules circuits électroniques capables de répondre instantanément aux sollicitations du microprocesseur.

MEMOIRES INTERNES, CENTRALES

On les appelle des « mémoires internes » (à l'ordinateur), ou encore des « mémoires centrales ».

En effet, il existe aussi des mémoires « externes », dites aussi « périphériques », telles que les cassettes magnétiques.

MEMOIRES EXTERNES, PERIPHERIQUES

Vous pouvez acquérir des programmes tout faits sur bande magnétique mais, pour pouvoir les exécuter, il vous faudra *obligatoirement* les transférer au préalable en « mémoire centrale » car ce n'est qu'à partir de celles-ci qu'ils pourront « tourner » (terme du jargon informatique signifiant « marcher »).

CELLULES MEMOIRES

Ces mémoires internes sont organisées en *cellules ;* chaque cellule est capable de mémoriser un caractère tel que ceux que vous frappez au clavier. Si vous vouliez écrire une lettre de 25 lignes de 60 caractères, par exemple, et la mettre en mémoire, il vous faudrait disposer de 25 fois 60 = 1 500 cellules en mémoire.

OCTET

CARACTERE

BYTE

La frappe d'un caractère se traduit, en effet, par l'émission d'une série de 8 impulsions électriques, ce qu'on appelle un *octet.* Or, une cellule est précisément capable de mémoriser 8 impulsions électriques, donc un octet. Notez qu'octet se dit « *byte* », en américain. Un total de 1 024 octets (et non 1 000 : ainsi le veut l'informatique !) s'appelle un *kilo-octet,* ou K octet (et en anglais « *kilobyte* »).

Rappelez-vous qu'en pratique, « octet » (ou « byte ») et « caractère » sont équivalents.

Le premier écran que vous fournit l'Oric – 1 vous précise de combien d'octets vous disposez, selon la version que vous avez acquise : 16 K ou 48 K. Ainsi, la version 48 K de l'auteur indiquait la disponibilité de 47 870 octets en affichant « 47 870 BYTES FREE », où *FREE* signifie *libre.* Cela fait un peu moins que 48 K exactement, car 48 K = 48 × 1 024 = 49 152 octets ; la différence est, en effet, utilisée par l'Oric – 1 lui-même pour ses diverses tâches internes.

Si vous deviez introduire en mémoire des programmes un peu longs, il vous faudrait vérifier qu'ils y trouveront suffisamment de place. Ainsi, n'employez pas une cassette de programme marquée 48 K avec un Oric – 1 de 16 K seulement. Par contre, l'inverse ne pose aucun problème : vous pourrez utiliser un programme de 16 K avec une mémoire centrale de 48 qui sera, alors, surabondante.

MEMOIRES VIVES

RAM

Il nous faut cependant encore vous donner une précision à propos des mémoires centrales. Sachez qu'on les classe en deux grandes catégories :

• *Les mémoires vives* dans lesquelles vous pouvez enregistrer des informations à volonté, et les lire, bien entendu. Elles sont affublées du sigle RAM (prononcez « *rame* »). C'est d'elles que nous avons traité ci-dessus, sans vous l'avoir précisé.

MEMOIRES MORTES

ROM

• *Les mémoires mortes :* vous ne pouvez plus rien y enregistrer car le fabricant l'a fait une fois pour toutes, et c'est définitif. Ces mémoires dites *mortes,* appelées aussi ROM (prononcez « *rome* »), contiennent des données qu'on ne peut modifier ; comme les pages imprimées d'un livre, on ne peut que les lire. Dans l'Oric – 1, elles stockent le langage Basic.

C'est bien pratique que les ROM soient indélébiles. Car les RAM, elles, ne conservent leur contenu *que pour autant que l'ordinateur reste sous tension.* Dès que vous le débranchez, ne serait-ce qu'une fraction d'instant, elles perdent les informations qu'elles stockaient.

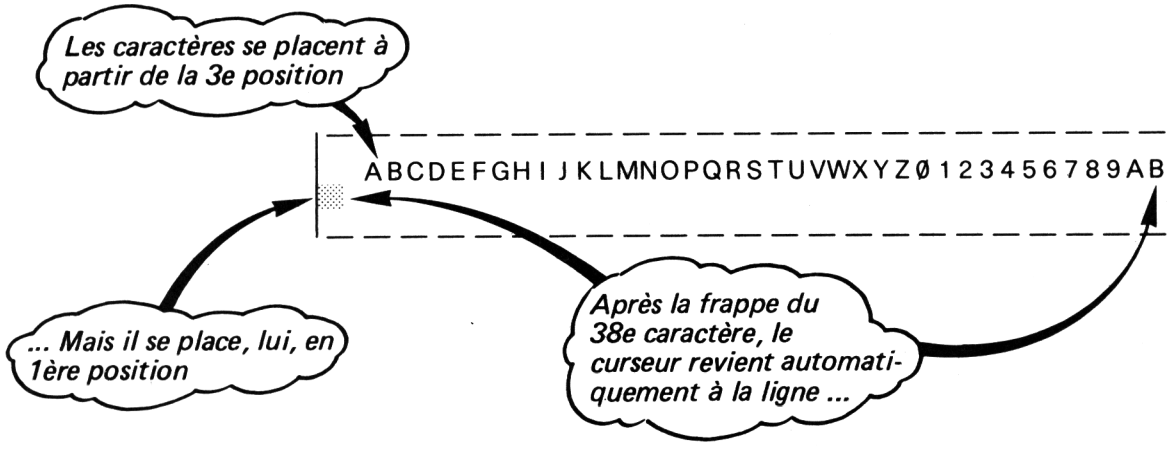
VOLATILITE DES MEMOIRES

Les mémoires vives sont affligées de ce dernier défaut, qu'on appelle la « volatilité ».

Vous devinez, maintenant, pourquoi on emploie des mémoires *externes,* encore appelées *mémoires périphériques,* telles que des cassettes de bande magnétique ? Les informations rangées sur bande magnétique peuvent être lues et relues à tout moment : elles ne sont pas "volatiles", elles. Lorsque vous remettrez votre micro-ordinateur sous tension, ses mémoires vives centrales ne contiendront plus rien ; vous pourrez alors les recharger avec le programme se trouvant sur support magnétique. Nous allons vous montrer comment dans le chapitre suivant.

5. Clavier et écran

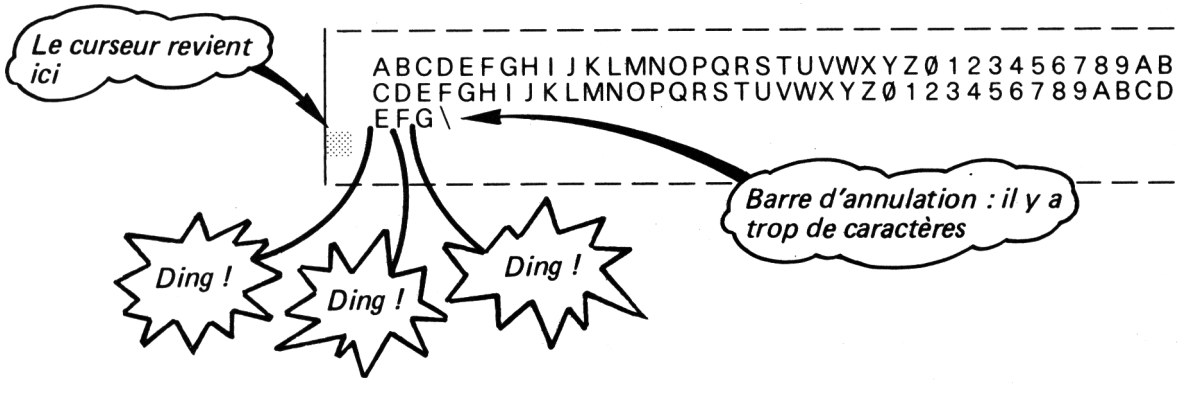
Revenons un moment sur le clavier et l'affichage. Votre Oric – 1 est-il en service et le téléviseur aussi ? Fort bien. Frappez donc l'alphabet, puis les chiffres de Ø à 9 (oui, nous allons barrer systématiquement le zéro), puis à nouveau l'alphabet : cela vous obligera à repérer l'emplacement des touches mais va aussi permettre de compter combien de caractères peuvent être inscrits sur une ligne :



La conclusion est évidente : une ligne d'écran contient 4Ø ''colonnes'' mais le texte n'en utilise que 38. Vous pouvez continuer à frapper la même séquence, par exemple : attention, l'Oric – 1 retournera à la ligne mais n'acceptera que 78 caractères répartis sur 3 lignes.

Dès le second retour à la ligne (76 caractères), sa sonnette vous prévient qu'il arrive au bout de ses possibilités d'accueil. Encore deux caractères, deux coups de sonnette, et ce sera tout !

Si vous vous entêtez et frappez un 79e caractère, il l'affichera mais le fera immédiatement suivre d'une barre oblique inversée signifiant qu'il annule la ligne, n'en tient pas compte :



Ceci nous permet de définir :

LIGNE PHYSIQUE
LIGNE LOGIQUE

- *Une ligne physique* de 38 caractères sur l'écran.
- *Une ligne logique* qui va, elle, jusqu'à 78 caractères répartis sur trois lignes.

Remarquez, maintenant, les quatre touches encadrant la barre d'espacement et marquées de flèches ←↓ ↑ et → et frappez-les : elles emmènent le curseur dans le sens indiqué. Notez que ←, contrairement à DEL, ne provoque pas l'effacement du caractère sur lequel le curseur passe ; vérifiez-le en emmenant le curseur sur du texte déjà frappé.

Expérimentez aussi les touches marquées de symboles ; en frappe *directe,* vous obtenez le symbole du bas ; avec SHIFT (maintenez SHIFT enfoncée et frappez la même touche), vous obtenez le symbole du haut.

CTRL

Mais peut-être voudriez-vous écrire en *minuscules* et non plus en *majuscules ?* Qu'à cela ne tienne, mais pour cela, il faut employer la touche CTRL (à gauche du clavier), qui signifie *contrôle* (ou plus exactement, *commande*). Elle s'utilise comme SHIFT : frappez-la et maintenez-la enfoncée, puis frappez sur une autre touche, et relâchez le tout.

MINUSCULES

MAJUSCULES

CTRL/T

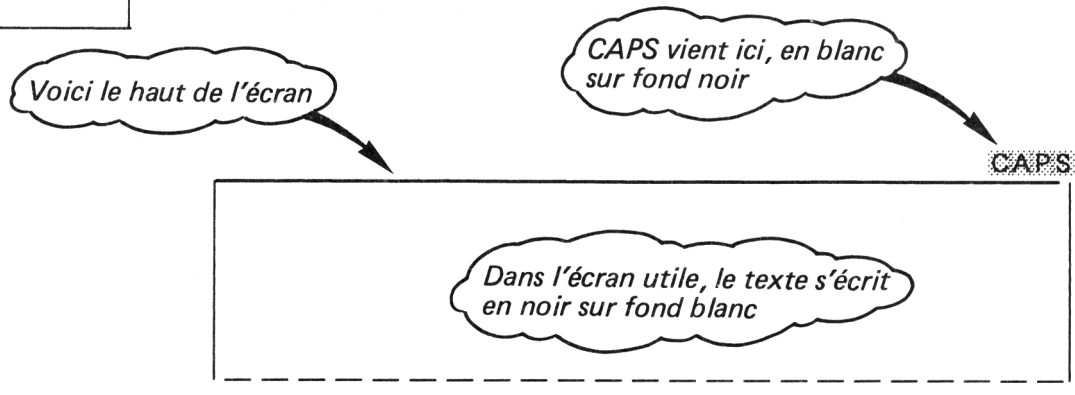
Pour passer aux minuscules, frappez CTRL et le T, ce que nous indiquerons par CTRL/T. Dès lors, tous les caractères alphabétiques vont s'inscrire en minuscules.

Désormais, votre clavier est semblable à celui d'une machine à écrire : toutes les frappes de touches alphabétiques donnent des minuscules. Si vous voulez passer aux majuscules, il faut frapper à la fois SHIFT et la lettre voulue. Les caractères autres que les lettres ne sont pas affectées.
Si vous voulez revenir au clavier « tout majuscules », il faut refrapper la même combinaison : CTRL/T.

Pour revenir aux majuscules, refrappez CTRL/T. Dès lors, tous les caractères alphabétiques viendront en majuscules. En plus, le haut de l'écran vous indiquera « CAPS ».

CAPS

Les lettres CAPS viennent pour « capitales », mot d'imprimerie signifiant majuscules. Cela donnera :



Le choix entre majuscules et minuscules dépend pour beaucoup de votre goût personnel. Mais toutes ces frappes n'ont-elles pas déjà bien rempli l'écran ? Si tel n'est pas encore le cas, continuez : vous constaterez que l'écran étant plein, une nouvelle ligne apparaissant en bas « chassera » la ligne du sommet qui disparaîtra, toutes les autres remontant d'un cran. Comptez alors les lignes *disponibles* (y compris celles qui ont pu être sautées) : il y en a exactement 27 sur l'écran « principal », plus la 28e qui vous indique « CAPS ».

FORMAT DE L'ECRAN

Le format de l'écran, en mode dit « texte », est ainsi de 28 lignes sur 40 colonnes.

La ligne supérieure, au-dessus du cadre principal, servira surtout pour l'affichage de messages divers.

RETURN

Nous vous parlerons ultérieurement de la touche ESC et de toutes les combinaisons possibles de CTRL, car nous en terminerons ici avec la touche la plus importante peut-être, celle marquée RETURN, soit *retour*. Elle correspond au *retour à la ligne* des machines à écrire mais surtout, c'est elle qui vous servira à dire à l'ordinateur « *Je viens de terminer un ordre ; prends-le en compte maintenant !* ». Pour cela, vous frapperez sur RETURN ; tant que vous ne l'aurez pas fait, l'Oric – 1 ne saura pas qu'il doit passer à exécution. Vous en apprécierez très vite l'usage. Si, en expérimentant le clavier, vous avez frappé sur RETURN, il est fort probable que le Basic de l'Oric – 1 n'aura rien compris à ce que vous avez voulu faire et vous aura retourné ce fameux message « ?SYNTAX ERROR » qui signifie à peu près « *Si vous voulez me commander, apprenez d'abord à le faire correctement !* »

QUESTIONS SUR LE CHAPITRE I

Vous pensiez en avoir terminé avec ce chapitre ? Que non : nous ne vous ferons grâce de rien, pas même de questions de cours ! En voici quelques-unes, qui évoquent des notions essentielles *traitées dans ce chapitre. Vous en trouverez d'ailleurs à la fin de tous les chapitres de ce livre.*

Dans tous les cas, ces questions resteront très simples et, pour y répondre, il vous suffira de cocher la case correspondante. Si vous marquez un doute, n'hésitez pas à relire le passage correspondant dans le chapitre. En voici donc une première série : cochez la case en face de la bonne réponse :

1. Le curseur sert à :
   - ☐ Exécuter des dessins sur l'écran
   - ☐ Marquer l'emplacement de la prochaine frappe

2. La touche DEL :
   - ☐ Efface le caractère précédent
   - ☐ Efface le caractère suivant

3. Pour utiliser la touche SHIFT, vous devez :
   - ☐ La frapper, la relâcher, puis frapper sur une autre touche
   - ☐ La maintenir enfoncée en frappant sur une autre touche

4. Dire d'une mémoire qu'elle est volatile signifie :
   - ☐ Qu'elle est comparable à un animal de basse-cour (en quoi ?...)
   - ☐ Qu'elle perd ses informations si on débranche l'ordinateur

5. Le format de l'écran, en mode texte, est de :
   - ☐ 28 lignes maximum sur 40 colonnes
   - ☐ 64 lignes sur 80 colonnes

Avez-vous bien répondu à tout ? Ce n'était pas bien difficile, avouez-le ! Vérifiez cependant vos réponses.

Réponses page 18

REPONSES AUX QUESTIONS SUR LE CHAPITRE I

1. Le curseur sert à marquer l'emplacement de la prochaine frappe.

2. La touche DEL efface le caractère *précédent*. Le curseur fait alors marche arrière.

3. Pour utiliser SHIFT, il faut maintenir cette touche enfoncée, puis en frapper une autre.

4. Une mémoire est dite *volatile* lorsqu'une coupure de l'alimentation lui fait perdre... la mémoire. Il n'y a là rien à voir avec des volatiles de basse-cour. La preuve ? *Volatil* s'écrit sans *e* au masculin (pour l'adjectif qualificatif).

5. Le format de l'écran, pour l'Oric – 1, est de 28 lignes sur 40 colonnes.

Combien d'erreurs avez-vous commises ? Zéro ? Tous nos compliments. Une ou deux : bien, bien, bien. Trois ou quatre : pourquoi ne pas relire ce chapitre ? Cinq : mais au fait, l'avez-vous bien lu une première fois ?

CHAPITRE II

MAGNÉTOPHONE ET CASSETTES DE PROGRAMMES

Disposez-vous de cassettes pré-enregistrées pour l'Oric ? Des jeux (« Les envahisseurs »...) ou des programmes d'enseignement (« Maths »...) ? Vous êtes immédiatement en mesure de les employer car pour cela, vous n'avez pas besoin d'en savoir davantage sur le fonctionnement de l'ordinateur. Vous allez, ainsi, pouvoir vous perfectionner en mathématiques, ou vous battre contre des soucoupes volantes.

C'est non seulement le « chargement » d'un programme qu'on va aborder ici, mais aussi la « sauvegarde », c'est-à-dire le transfert sur bande magnétique (non « volatile ») d'un programme que vous avez créé, par exemple.
Le mode d'emploi de ce chapitre est ainsi le suivant : lisez immédiatement sa première partie mais, parce qu'on n'a pas encore étudié la programmation, sautez la seconde. Vous y reviendrez par la suite, dès que le besoin s'en fera sentir, c'est-à-dire dès que vous aurez des programmes à « sauvegarder ».

Principaux thèmes étudiés

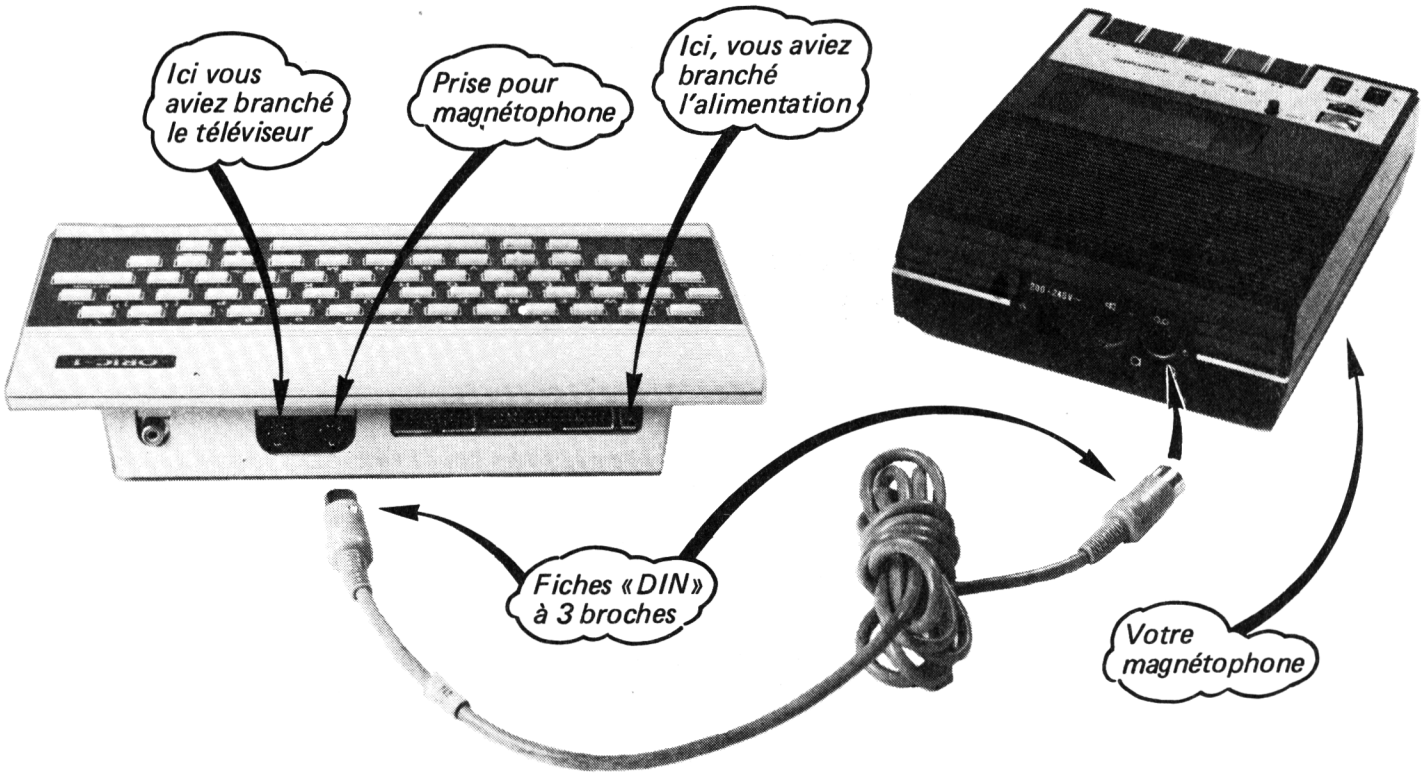
Connexion d'un magnétophone
Chargement d'un programme
CLOAD
Vitesse lente
,S
Ré-initialisation
RESET
Démarrage à chaud
Démarrage à froid
Sauvegarde d'un programme
CSAVE
Démarrage automatique
CSAVE AUTO

1. Le magnétophone

Votre Oric utilise des cassettes tout à fait standards, telles que celles sur lesquelles vous enregistrez de la musique. Tout au plus conseillera-t-on alors d'utiliser des bandes de bonne qualité.

CONNEXION D'UN MAGNETOPHONE

Le magnétophone sera votre magnétophone habituel. Les appareils très bon marché donnent d'excellents résultats. Il vous faut cependant connecter le magnétophone à l'Oric et pour cela, nous vous conseillons d'employer le câble que vous fournit le fabricant. C'est un câble terminé à ses deux extrémités par des fiches rondes (aux normes « DIN »), à trois broches mâles. Insérez l'une de ces fiches dans la prise femelle à 7 broches, elle, située à l'arrière de l'Oric, la seconde dans la prise femelle correspondante de votre magnétophone, qu'elle soit à 3 ou 5 broches, peu importe.



L'Oric, vu de derrière, afin de vous montrer où connecter le magnétophone.

Méfiez-vous des cordons qu'on trouve dans le commerce, avec cinq broches au lieu de trois ; dans ce cas, en effet, des broches sont souvent court-circuitées et vous éprouverez des surprises les plus diverses (l'auteur en a fait l'expérience et a perdu le son...).

Par contre, si votre magnétophone est muni d'une entrée de commande à distance (mise en marche et arrêt du moteur), n'hésitez pas à employer un connecteur 7 broches, branché comme la notice de l'Oric le décrit. Vous vous épargnerez ainsi la peine de mettre le magnétophone en marche puis de l'arrêter, votre ordinateur s'en chargeant pour vous. Dans ce qui suit, nous allons supposer que vous ne disposez pas de télécommande et nous allons vous imposer cette « corvée » en plus.

Sur le magnétophone lui-même, vous n'aurez qu'à ajuster le volume sonore, qui ne nous a pas paru critique du tout. L'auteur a obtenu d'excellents résultats en promenant le bouton de volume sonore pratiquement sur tous les réglages possibles. Réglez-vous cependant, par souci de sécurité, vers le milieu de la gamme ou un peu au-delà ; nous sommes restés sur le réglage 6 pour un bouton gradué de Ø à 1Ø.

Un conseil : les têtes de lecture-écriture de votre magnétophone doivent être propres : nettoyez-les de temps à autre.

2. Le chargement d'un programme

On vous a déjà dit que les mémoires internes (vives) des ordinateurs étaient "volatiles". Lorsque vous mettrez votre Oric en service, ses mémoires vives ne contiendront rien. Par contre, vous disposez peut-être d'une cassette pré-enregistrée.

Si votre Oric est une version 16 K, il faudra utiliser une cassette contenant un programme ne dépassant pas 16 K. Vérifiez sur votre cassette si elle est destinée à une version 16 K (ou 48 K) : c'est généralement indiqué très lisiblement.

Une fois sélectionnée la cassette voulue, on va transférer en mémoire centrale le programme qu'elle contient.

CHARGEMENT D'UN PROGRAMME

Cette opération s'appelle le chargement. Vous l'exécuterez chaque fois que vous voudrez employer un autre programme à partir d'une mémoire périphérique (d'une cassette, dans notre cas).

Placez la cassette dans le magnétophone. Très souvent et par sécurité, une cassette est enregistrée sur ses deux faces *avec le même programme ;* vous pourrez donc employer indifféremment n'importe laquelle. Rembobinez la cassette, puis respectez les instructions suivantes :

CLOAD

1 — Frappez au clavier le mot CLOAD. Il provient de l'anglais *LOAD,* pour *chargement,* précédé par un C signifiant *cassette.*

2 — Si votre cassette ne contient qu'un seul et unique programme, frappez ensuite *deux fois* des guillemets après CLOAD (vous pouvez laisser un espace entre CLOAD et la première paire de guillemets).

Vous obtiendrez les guillemets avec la touche SHIFT.

En réalité, l'ordre CLOAD complet demande qu'on précise le *nom du programme,* écrit entre guillemets. Celui-ci est inscrit sur la cassette. Si vous n'indiquez pas de nom, l'Oric va charger le premier programme qu'il rencontrera. S'il n'y en a qu'un, c'est forcément le bon.
Vous pouvez, cependant, indiquer le nom ; deux cas se présentent ici :

• *La bande comporte plusieurs programmes.* En indiquant le nom de celui que vous voulez charger, l'Oric le recherchera et ne chargera que celui-ci.

• *Elle ne contient qu'un seul programme :* ce sera une vérification.

Attention : si vous frappez le nom, respectez scrupuleusement son orthographe et même les espaces.

La moindre erreur serait interprétée par l'Oric comme un autre nom, et l'ordinateur ne retrouverait pas le programme voulu.

3 — Maintenant, vérifiez sur la cassette si ce nom n'est pas suivi par une virgule, puis par la lettre S. Ce sera très généralement le cas. Frappez alors une virgule puis un S majuscule.

VITESSE LENTE
,S

L'Oric dispose de deux vitesses pour enregistrer et lire des cassettes, une rapide et une lente. La lente fournit des résultats bien plus fiables, d'où son emploi. « Lente » se dit « slow », en anglais, d'où le S.

Si vous ne frappiez ni la virgule, ni le S, l'Oric passerait la grande vitesse... et bien sûr, n'aurait plus le temps de lire une bande enregistrée en basse vitesse.
En supposant que vous veniez de mettre votre ordinateur en service, l'écran devrait afficher :

Ça, c'est pour un 48 K. Pour un 16 K, ce sera moins

```
ORIC EXTENDED BASIC V1.Ø
C 1983 TANGERINE

47870 BYTES FREE

Ready
CLOAD "",S
```

Vous venez de frapper ceci

Le curseur est ici

Notez que nous n'utilisons pas de nom de programme, dans les guillemets, pour une autre raison « nécessaire et suffisante » : nous ignorons lequel vous allez employer.

4 — Maintenant, frappez sur la touche marquée RETURN.

C'est la touche RETURN qui indique au Basic qu'il doit acquérir l'ordre frappé, puis l'exécuter. Sans RETURN, rien ne se passera.

Aussitôt, l'ordinateur part à la recherche du programme, ce qu'il a l'obligeance de vous indiquer sur sa 28e ligne (celle du haut) :

```
SEARCHING ....
```

«Je cherche ...» : c'est écrit en blanc sur fond noir

5 — Mais il ne peut rien trouver, car vous n'avez pas encore mis le magnétophone en marche, en mode « lecture ». Faites-le donc.
Dès lors, le Basic « lit » la bande ; dès que le programme est trouvé, le message change et devient :

```
LOADING ....
```

«J'ai trouvé le programme et je suis en train de le charger...»

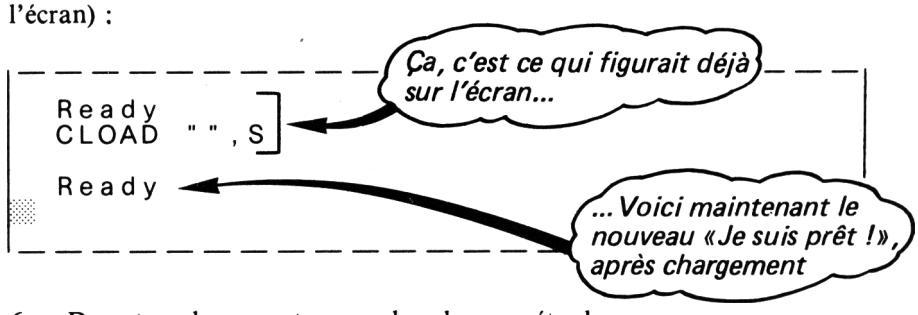
Si vous avez donné le nom du programme, ce même message est suivi de ce nom, par exemple "DEMONSTRATION" :

```
LOADING ... DEMONSTRATION
```

Armez-vous de patience : pour un programme destiné à une mémoire de 48 K et en mode lent (S), il faudra cinq bonnes minutes. A l'issue du chargement :

• *le programme démarrera automatiquement.* Le plus souvent alors, c'est lui qui vous indiquera ce que vous devrez faire. Par exemple et pour lutter contre des soucoupes volantes, il vous dira que vous devez frapper sur la barre d'espacement pour envoyer le faisceau laser contre les envahisseurs.

• sinon, l'Oric vous indiquera que le chargement est terminé en affichant son message habituel READY, ce qui donnera (si l'on ne retient qu'une fraction de l'écran) :



6 — Dans tous les cas, stoppez alors le magnétophone.

7 — Si le programme chargé ne démarre pas automatiquement, frappez les trois lettres RUN, ce qui signifie "En marche", en anglais, puis frappez la touche RETURN et il sera "lancé".

En cas d'erreur, n'hésitez pas à recommencer toutes ces opérations.

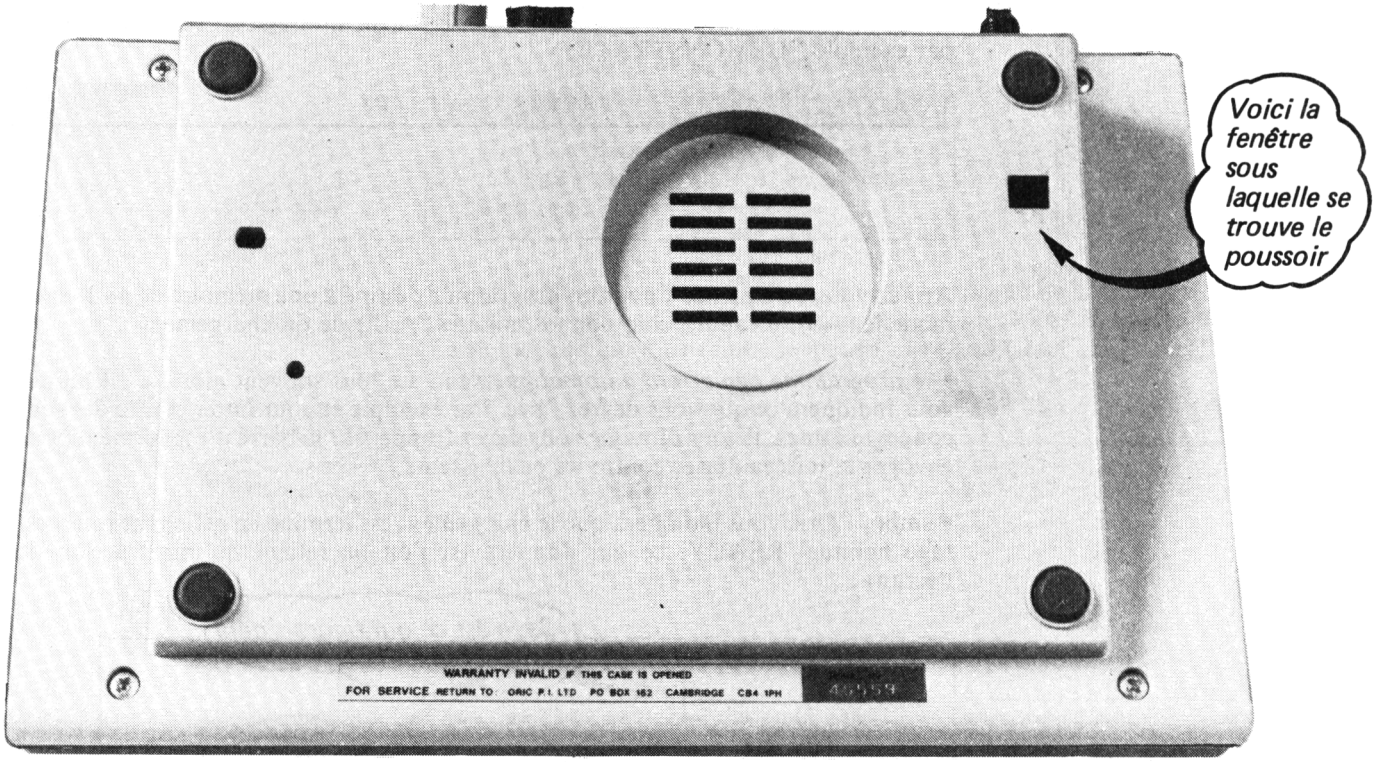
Si vous frappez mal l'ordre, vous obtiendrez peut-être le message ?SYNTAX ERROR. Recommencez alors. Ou bien vous constaterez que toute la bande a été lue et que le message sur la ligne supérieure est toujours "SEARCHING..." (je cherche...) ; demandez-vous alors si vous n'avez pas omis de connecter le magnétophone au micro-ordinateur.

L'ordinateur est têtu : après un ordre de chargement, il cherche un programme et le veut ! Tant qu'il ne l'aura pas trouvé, il continuera à chercher sans accepter d'autres ordres.

RE-INITIALISATION

Le clavier est alors "inerte", ne répond plus à vos frappes. Pour pouvoir recommencer et passer de nouveaux ordres, débranchez un instant l'ordinateur, puis remettez-le sous tension, ce qui va le *ré-initialiser,* comme l'on dit : vous retrouverez le premier écran.

Il existe toutefois une méthode plus élégante pour ré-initialiser l'ordinateur, mais elle n'est pas plus pratique, vous allez en juger : retourner complètement l'Oric, sens dessus-dessous. A l'extrémité droite du boîtier se trouve une petite "fenêtre" rectangulaire en face de laquelle a été placé un bouton-poussoir (qu'on ne distingue pas). Introduisez un crayon dans cette fenêtre et poussez le bouton.



Pour ré-initialiser la machine, vous pouvez agir sur le poussoir accessible par le dessous de l'Oric.

Aussitôt, l'écran s'efface et vous obtenez :

```
Ready
```

Dès lors, vous pouvez recommencer.

RESET

Cette action s'appelle le RESET (remise à zéro). Le RESET ne fait pas perdre à la mémoire centrale ce qu'elle pouvait déjà contenir, contrairement à une interruption de l'alimentation.

Servez-vous du RESET chaque fois que vous perdez le contrôle de l'ordinateur. Cela vous sera très utile, surtout lorsque vous programmerez.

DEMARRAGE A CHAUD, A FROID

Le RESET est encore appelé un « démarrage à chaud ». Le « démarrage à froid » est celui qu'on provoque en remettant l'Oric sous tension.

Nous supposons maintenant que votre programme « tourne » et que vous êtes en train d'essayer de poser votre super-avion 787 sur la piste d'atterrissage. L'auteur souhaite que vous réussissiez mieux que lui · il a « crashé » l'avion, ce qui lui a valu en cours de manœuvre le message *« Attention, cet avion est encore sous garantie ! »*.

3. Sauvegarde d'un programme

Nous allons vous montrer, maintenant, comment transférer sur bande magnétique un programme se trouvant en mémoire centrale. Ce sera soit un programme que vous venez de créer, soit tout autre programme que vous avez précédemment chargé en mémoire centrale à partir d'une cassette et dont vous voulez faire une copie sur une autre cassette.

Si vous ne faites qu'aborder ce livre, vous n'avez pas encore commencé à programmer ; dans ce cas, nous vous conseillons de sauter ce qui suit. En effet, nous allons faire appel à quelques notions qui ne seront étudiées que dans les chapitres ultérieurs (par exemple, les ordres PRINT, LIST...). Dès que vous aurez débuté la programmation, vous reviendrez à ce texte ; nous vous indiquerons à quel moment. En effet, nous avons jugé préférable de regrouper en un unique chapitre tout ce qui a trait aux cassettes.

SAUVEGARDE D'UN PROGRAMME

Le rangement sur bande magnétique d'un programme se trouvant en mémoire centrale s'appelle la « sauvegarde ».

Créons un programme ultra-simple afin de le sauvegarder (c'est inutile si vous avez votre propre programme en mémoire centrale : servez-vous alors de celui-là). Ce programme comporte deux lignes :

```
Ready
10 PRINT "BONJOUR"
20 END
```

Afin de le sauvegarder sur cassette, vous pouvez ou non lui décerner un nom, obligatoirement écrit « entre guillemets ».

Si vous ne lui attribuez pas de nom, il faudra quand même frapper les guillemets ⊓⊓ mais sans rien écrire dedans, sans même placer un espace à l'intérieur : le Basic considérerait cet espace comme un nom !

CSAVE

Appelons ce programme « ADIEU ». L'ordre de sauvegarder s'écrit CSAVE, avec SAVE pour *sauvegarder* précédé de C, pour *cassette*. L'ordre complet serait donc :

```
CSAVE "ADIEU"
```

Le curseur est là : vous n'avez pas encore frappé sur RETURN, donc l'ordre de sauvegarde n'est pas encore exécuté

Vous pourriez dès lors lancer l'enregistrement : celui-ci se fera très rapidement, à la vitesse de 2 400 « bauds », ce qui correspond à quelque 200 caractères par seconde. Pour travailler à cette relativement grande vitesse, il faut un matériel impeccable : magnétophone (ses têtes, surtout) et bande. Comme, en général, tel n'est pas le cas,

nous vous conseillons par sécurité d'adopter la vitesse « lente », de 300 bauds, donc 8 fois inférieure. Encore faut-il indiquer au Basic qu'on change de vitesse. Or, le mot « lent » se dit « *slow* », en anglais ; vous ajouterez donc à la suite de l'ordre *une virgule,* puis un S, ce qui donne :

```
CSAVE "ADIEU",S
```

Si vous ne voulez pas mettre de nom, écrivez simplement :

```
CSAVE "",S
```

Attention : pas d'espace dans les guillemets

Par contre, vous pouvez mettre un espace après les guillemets si vous le désirez, pour le coup d'œil

Dès lors, le processus est le suivant :

1. Mettez une cassette (vierge, de préférence) dans le magnétophone.

2. Rembobinez-la, remettez le compteur à zéro si votre magnétophone en possède un. Vous pouvez enregistrer n'importe où sur la bande mais n'oubliez pas de noter alors l'indication du compteur.
Attention : toutes les bandes commencent par une petite amorce sur laquelle on ne peut rien enregistrer.

3. Mettez le magnétophone en marche en position *enregistrement* (et non lecture : ne vous trompez pas). Laissez passer éventuellement l'amorce, puis frappez sur RETURN. L'ordinateur va alors exécuter l'ordre ; il commence par émettre un signal servant à stabiliser le fonctionnement du magnétophone, puis envoie la salve des ondes traduisant votre programme. Pendant le court instant que cela dure, vous lirez sur la ligne supérieure de l'écran ;

```
SAVING
```

ou alors, si vous avez donné le nom du programme :

```
SAVING ADIEU
```

Dès que le programme est sauvegardé, l'écran ré-affiche un READY :

```
CSAVE "",S
Ready
```

4. Stoppez alors le magnétophone car la sauvegarde est terminée.

5. Indiquez, sur la cassette, *le nom* du programme, *où il commence* et *où il finit* sur la bande, et indiquez enfin la vitesse. Par exemple, ajoutez « ,S » pour préciser qu'il s'agit de la vitesse lente.

N'omettez jamais de porter sur vos cassettes toutes les indications utiles. Vous ne vous énerverez pas, ensuite, à rechercher « quelque part » un programme donné.

Au cours de la sauvegarde et selon le type de magnétophone que vous possédez, vous entendrez ou non la « modulation », les trains d'ondes qui sont transférés. Le bruit en est plutôt désagréable ! Il en ira de même, d'ailleurs, lors de l'opération inverse, le chargement.

6. Voulez-vous prendre une précaution supplémentaire ? Sauvegardez *une seconde fois* ce même programme.

Cela étant fait, on va vérifier que tout a bien fonctionné.

4. Les règles du chargement

Afin de vérifier que la sauvegarde s'est correctement déroulée, on va effacer la mémoire centrale, afin de recharger ce programme.

Pour effacer la mémoire centrale, utilisez l'ordre NEW qui indique à l'ordinateur qu'on veut faire le ménage dans la mémoire, la vider, afin de charger un *nouveau* programme. Frappez NEW, puis RETURN :

```
NEW
Ready
```

Ici, vous avez fait RETURN

Par acquit de conscience, vérifiez que la mémoire est bien vide en demandant le *listage* de ce qu'elle contient, ce qui se commande LIST ; n'oubliez pas le RETURN :

```
LIST

Ready
```

Aucun listage n'apparait car il n'y a plus rien en mémoire

Le processus de chargement est exactement le même que celui qu'on a vu au début du chapitre. Rembobinez la bande et placez-vous au point de départ du programme, puis frappez l'ordre :

```
CLOAD "",S
```

Vous avez fait RETURN ici

ou encore, si vous voulez donner le nom :

```
CLOAD "ADIEU",S
```

Il faut obligatoirement charger un programme avec la même vitesse qu'on a utilisée pour sa sauvegarde. Ce sera donc la vitesse lente, indiquée par le suffixe « ,S ».

Dès le RETURN, le curseur revient à la ligne et la 28e ligne, celle au-dessus de l'écran habituel, indique « SEARCHING... » (« je cherche le programme »). Démarrez alors le magnétophone en position *lecture* (et non enregistrement).

Notez bien cette différence :
• A la sauvegarde, on démarre d'abord le magnétophone avant de frapper RETURN.
• Au chargement, on frappe d'abord RETURN, puis on lance le magnétophone.

L'ordinateur rencontrant le programme va afficher, tout en haut de l'écran, « LOADING... » suivi du nom de ce programme si vous l'avez indiqué, et à l'issue du chargement, il ré-affichera un *Ready* sur l'écran. Stoppez alors le magnétophone.

Si vous avez fourni le nom du programme, l'ordinateur ne chargera pas le premier qu'il rencontre mais uniquement celui dont le nom est en conformité absolue avec celui que vous avez donné.

Il est peut-être dommage que l'Oric – 1 n'affiche pas, au passage, les noms des programmes qu'il croise et qui ne sont pas demandés. Mais on ne peut pas avoir toutes les qualités.
Bref, le programme se trouve à nouveau en mémoire centrale. Vérifions-le en faisant un LIST, par exemple :

A la fin du chargement l'Oric affiche ce «prêt !»

```
CLOAD "",S

Ready
LIST

10 PRINT "BONJOUR"
20 END

Ready
```

Voici le listage exact du programme que vous aviez sauvegardé

Ce listage est parfaitement correct. Vous auriez aussi pu lancer ce programme avec RUN, ce qui aurait donné :

```
Ready
RUN
BONJOUR

Ready
```

En effet, le RUN affiche «Bonjour»

DEMARRAGE AUTOMATIQUE
AUTO

Il est même possible de faire automatiquement démarrer un programme à l'issue de son chargement, simplement en ajoutant à la fin de l'ordre *de sauvegarde* le mot AUTO. Faites-en l'expérience : puisque votre programme est en mémoire centrale, sauvegardez-le à nouveau mais en faisant, cette fois (repérez bien où vous l'enregistrez sur la bande) :

```
Ready
CSAVE  " " , S , AUTO
```

Ou encore, ce qui est rigoureusement identique quant à l'effet :

```
Ready
CSAVE  " " , AUTO , S
```

Puis, effacez la mémoire centrale avec NEW, et rechargez le programme. Automatiquement, il démarrera, ce qui donnera :

```
Ready
CLOAD  " " , S
BONJOUR

Ready
```

Aussitôt chargé, le programme est exécuté

Vous comprenez, maintenant, pourquoi les cassettes pré-enregistrées vous offrent souvent des programmes à auto-démarrage : elles ont été enregistrées avec cet ordre AUTO.

Ajoutons encore que tous les chargements ne seront pas forcément réussis. Dans ce cas, l'Oric vous préviendra en affichant, après l'ordre de chargement :

```
CLOAD  " " , S

FILE ERROR / LOAD ABORTED
Ready
```

«Il y a une erreur quelque part : j'ai stoppé le chargement !»

Il ne vous restera plus, alors, qu'à recommencer, peut-être en essayant la seconde copie du même programme si l'on suppose que la bande puisse être, en un passage quelconque, défectueuse.

QUESTIONS SUR LE CHAPITRE II

Ici, exceptionnellement, vous n'aurez pas droit à nos traditionnelles questions. Jouez plutôt aux « Envahisseurs », par exemple (et, s'il vous plaît, ne restez pas forcément sur les vitesses de jeu pour débutants !).
N'oubliez pas que le jeu est une activité intéressante à de nombreux points de vue, ne serait-ce que pour bien se familiariser avec l'ordinateur.

REPONSES AUX QUESTIONS SUR LE CHAPITRE III

1.PRINT est un ordre d'affichage sur l'écran.

2. PRINT "3 + 2" affiche 3 + 2 (reproduction, puisqu'il y a des guillemets).

3. PRINT 3 + 2 affiche 5 (pas de guillemets, donc exécution de l'opération).

4. PRINT "3 + 2 affiche 3 + 2. Bon enfant, le Basic accepte que vous ne refermiez pas les guillemets si vous faites un RETURN.

5. Le symbole ? remplace le mot PRINT. Donc, vous afficherez 5 ; il n'y a pas d'erreur.

6. Par contre, il y en a une, ici, car on a confondu le lettre O avec le chiffre zéro. La machine s'en sort en additionnant 1Ø + 1 et affiche 11, puis s'occupe de la lettre O qu'elle considère comme une variable à laquelle on n'a rien affecté encore, et lui décerne un zéro !

7. Rappelez-vous : le symbole de la multiplication n'est pas le X mais l'astérisque (*). La machine répondra 7 Ø car pour elle X3 est aussi le nom d'une variable qui n'a reçu aucune affectation.

8. C'est une élévation à la puissance 3 ; vous lirez donc 27.

9. L'écran affichera 9/3 = 6.

1Ø. Dans un ordre PRINT, la virgule sert *à la fois* à séparer des constantes et à introduire 4 espaces dans leur affichage. Les deux réponses sont bonnes.

11. A$ est un nom de variable chaîne, grâce au symbole $

12. Oui ; en fin de course, T vaut 1.

CHAPITRE III

L'ORDINATEUR EN « MODE DIRECT »

Voici le premier chapitre consacré à la programmation, dans ce qu'on appelle le « mode direct ». On donne un ordre à l'ordinateur et, dès la frappe de « RETURN », il l'exécute. On a déjà exploité ce mode direct avec les cassettes, aussi va-t-on en profiter pour examiner, ici, d'autres instructions fondamentales. Le chapitre suivant abordera le mode « programme ».

Principaux thèmes étudiés

Guillemets	Elévation à puissance	Variable numérique
PRINT	Racine carrée	Constante chaîne
RETURN	SQR	Variable chaîne
Syntaxe	Virgule dans un PRINT	Affectation
Annulation d'une ligne	Effacement de l'écran	Symbole d'affectation (=)
CTRL/X	CTRL/L	Noms de variables
Point d'interrogation pour PRINT	CTRL/H	Mots réservés
Mode direct	CTRL/I	Chaînes de caractères
Rôle des guillemets	CTRL/J	Chaîne vide
Point-virgule	CTRL/K	Concaténation de chaînes
Zéro et lettre O	CTRL/M	Symbole + de concaténation
Multiplication (*)	Chaînes de caractères	Sonorisation des touches
Division (/)	Symbole $	CTRL/F
Point décimal	Constante	Effacement d'une ligne
Virgule décimale	Variable	CTRL/N
	Constante numérique	

1. L'instruction PRINT

Avec ce chapitre, nous abordons la programmation de l'Oric – 1, dans ce langage qu'on appelle le Basic et dont nous avons parlé au début de ce livre.

GUILLEMETS

Commencez donc par frapper au clavier les mots suivants, en respectant rigoureusement cette façon de les écrire ; cela signifie qu'il faudra placer des espaces comme ils figurent ici, pas davantage (après tout, il sera toujours temps d'essayer après), ainsi que des guillemets. Les guillemets sont obtenus *avec* la touche SHIFT : maintenez SHIFT enfoncée, puis frappez sur la touche marquée de guillemets, et relâchez le tout.

Rappelez-vous aussi que l'espace arrière, marqué DEL, sert à effacer le caractère précédent.

Voici donc ce que vous devez frapper et ce qui apparaîtra sur l'écran, à supposer que vous veniez de mettre votre ordinateur en service ; le mot nouveau, un ordre du Basic, est PRINT, ici ; respectez bien son orthographe :

```
ORIC EXTENDED BASIC V1.Ø
C 1983 TANGERINE

47870 BYTES FREE

Ready
PRINT "BONJOUR"
```

Frappez cette ligne

Le curseur vient ici

Mettez le mot BONJOUR «entre guillemets»

PRINT

Le mot PRINT est un mot anglais signifiant « imprimer ». On l'utilisait aux âges héroïques de l'informatique, quand les écrans étaient rares et chers. On l'a conservé avec les écrans mais dans l'acception *afficher*. Cet ordre Basic ordonne à la machine d'afficher tout ce qui est encadré par les guillemets ; c'est, aussi, la première instruction Basic que vous écrivez. Félicitations !

RETURN

Pour l'instant, l'ordre figure sur l'écran mais n'a pas encore été acquis par le Basic, ni donc exécuté. Il manque encore cette indication qu'on pourrait traduire par « *A vous de jouer, Basic : prenez cet ordre et faites ce que vous avez à faire !* ». Tout cela s'exprime très simplement : il suffit de frapper sur la touche RETURN (« retour »).

La frappe de RETURN fait prendre l'ordre en compte par l'ordinateur. Il faudra la frapper après un ordre, une instruction.

On verra que RETURN sert aussi à ramener le curseur à la ligne, au début de la ligne suivante, d'où le nom qui lui a été attribué. Frappez donc sur RETURN ; aussitôt, le Basic obéissant affiche sur la ligne suivante BONJOUR, puis saute une ligne et vous indique qu'il est de nouveau prêt, ce qui donne :

```
PRINT "BONJOUR"
BONJOUR

READY
```

Ceci c'est le Basic qui l'a affiché, en exécution de votre ordre

la machine attend maintenant la suite

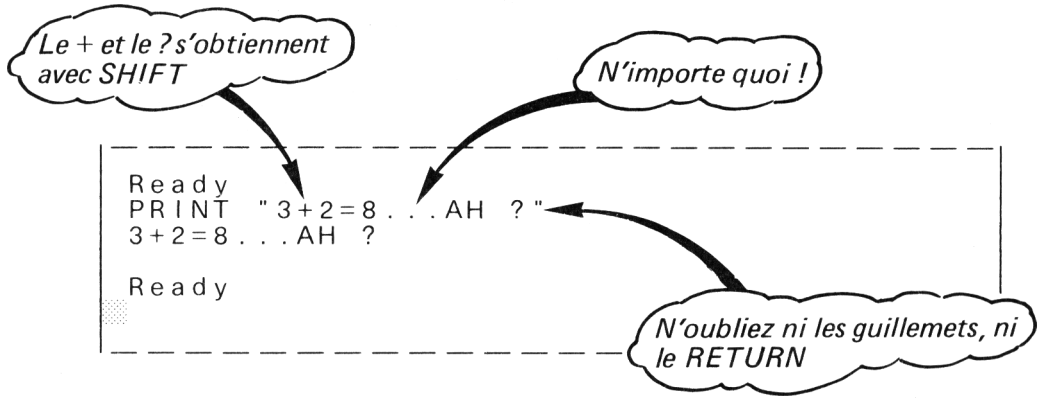
Essayons maintenant un message plus compliqué ; frappez d'abord la ligne indiquée puis faites RETURN, et vous obtiendrez :

N'oubliez pas les guillemets...

```
Ready
PRINT "PARIS, LE 14 JUILLET 1789"
PARIS, LE 14 JUILLET 1789

Ready
```

... Et pensez à frapper ici RETURN

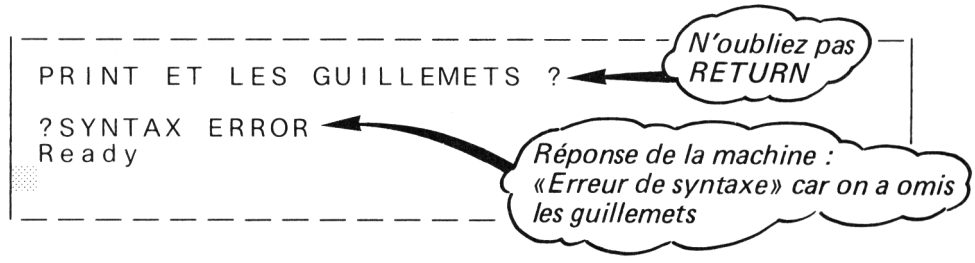
A nouveau, la machine a reproduit fidèlement votre message. Il peut être absolument quelconque, à la condition d'être encadré par des guillemets ; par exemple :



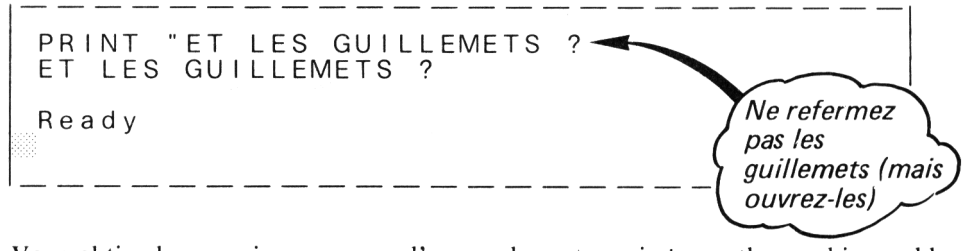
SYNTAXE

Sachez que la machine ne s'intéresse pas au sens (au contenu) du message, à la « sémantique », mais à sa forme, sa « syntaxe ».

Ainsi, les guillemets lui sont nécessaires. Si vous les oubliez, elle ne comprendra plus et vous renverra un message d'erreur :



A la limite, elle acceptera que vous ne *refermiez* pas les guillemets si vous faites un RETURN :



Vous obtiendrez aussi un message d'erreur de syntaxe si vous orthographiez mal le mot print, ou introduisiez des espaces entre ses lettres, ou l'écriviez en minuscules : vérifiez-le.

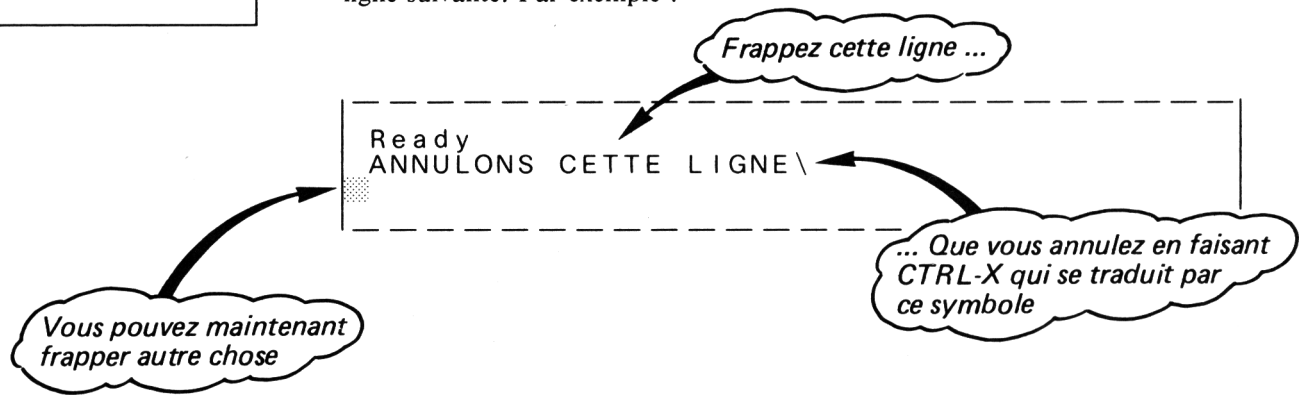
Les mots de commandes doivent être frappés en majuscules. Ce que vous mettez dans les guillemets peut être en minuscules ou majuscules, peu importe, l'ordinateur respectera votre choix de caractères.

Rappelez-vous qu'on passe des majuscules aux minuscules, et vice-versa, avec CTRL/T (maintenez CTRL enfoncée et frappez sur T). Après le premier retour aux majuscules, le mot CAPS (pour « capitales », ce qui signifie *majuscules* en termes d'imprimerie), apparaît en haut et à droite de l'écran. Il disparaît lorsque vous revenez aux minuscules. En mode *minuscules,* vous obtiendrez des majuscules avec la touche SHIFT, comme sur un clavier de machine à écrire.

ANNULATION D'UNE LIGNE

CTRL/X

Si vous commettez trop de fautes dans une ligne, n'hésitez pas à frapper RETURN et à la recommencer ; tant pis pour le message d'erreur : vous en obtiendrez bien d'autres, vous le verrez. Une autre méthode, plus élégante, consiste à frapper au bout d'une ligne qu'on veut annuler (et *avant* d'avoir fait RETURN) le code CTRL/X (maintenez CTRL enfoncée et frappez X). Le Basic place alors une barre oblique inversée au bout de la ligne, l'annule, et ramène le curseur au début de la ligne suivante. Par exemple :

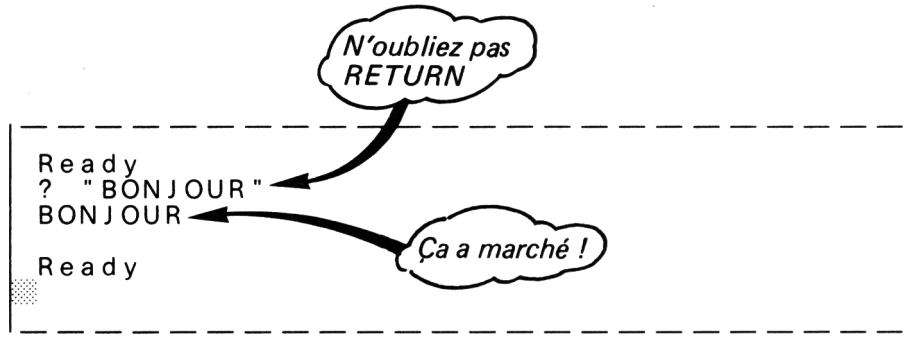


Dans la ligne ci-dessus, on n'avait même pas mis d'ordre, tel que PRINT. C'est lassant, aussi, de toujours frapper ces cinq lettres... Bon, d'accord : remplacez-les par un simple point d'interrogation.

POINT D'INTERROGATION POUR PRINT

Dans le code du Basic, un point d'interrogation peut remplacer le mot PRINT.

Vérifions-le avec cet exemple :



On utilisera ce point d'interrogation qui permet d'économiser 4 frappes...
A propos : vous devez maintenant avoir acquis l'automatisme de la frappe sur RETURN, et ce à la fin de chaque ordre. Aussi ne vous le rappellera-t-on plus systématiquement, sauf lorsque c'est nécessaire à nos explications. D'autre part, vous avez pu constater que le mot « *Ready* » revient constamment ; nous ne le reproduirons plus sur nos «'écrans » sauf, à nouveau, lorsqu'il se révèlera intéressant à placer.

Enfin, nous vous proposons une convention d'écriture dans nos « écrans » : tout ce que vous devez frapper sera indiqué en gras, alors que les réponses de la machine ou ses messages resteront en maigre.

Rappelez-vous qu'il ne s'agit que d'une convention, puisqu'en réalité, tous les caractères auront le même aspect sur votre téléviseur.

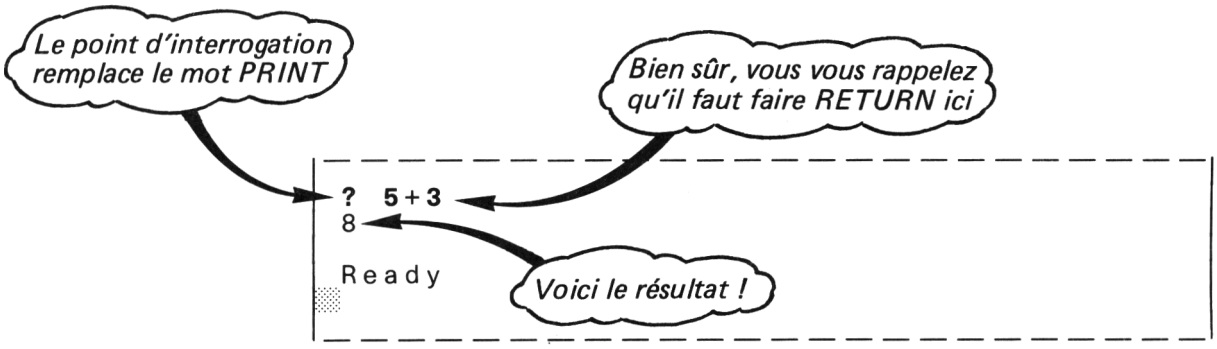
2. Premiers calculs

MODE DIRECT

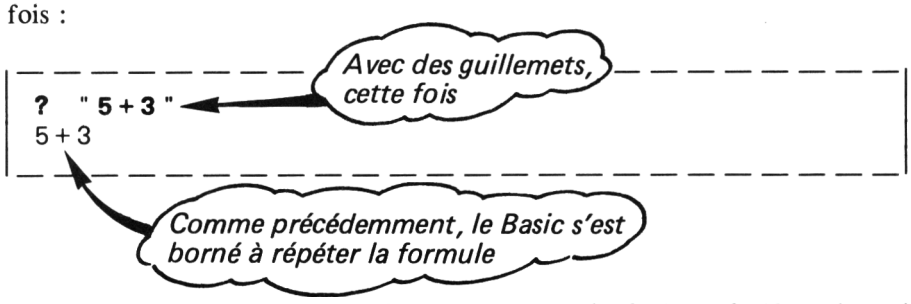
Vous avez pu constater que, lorsqu'on frappait un ordre au clavier suivi par un RETURN, la machine l'exécutait *aussitôt*. C'est cela qu'on appelle le *mode direct*.

> **En mode direct, un ordre est aussitôt exécuté (après RETURN). On appelle encore ce mode « mode immédiat », ou « mode conversationnel ».**

Si l'on passe aux chiffres, on l'appellera même « mode calculette », voici pourquoi. Frappez l'addition suivante, *sans placer de guillemets,* et faites RETURN ; surprise : vous n'obtiendrez pas de message d'erreur mais le résultat ! Voici ce que cela donne :



Nous avons écrit **en gras** ce que vous deviez frapper, selon nos conventions. La machine a répondu 8 ; heureusement ! Recommencez avec des guillemets, cette fois :



ROLE DES GUILLEMETS

Ainsi, on peut demander au Basic soit de reproduire la formule (dans des guillemets), soit d'exécuter le calcul (sans guillemets). Combinons ces deux modes, en séparant les deux codes par un point-vigurle :

```
? "5+3"="  ;  5+3
5+3=8
```

La première expression entre guillemets est reproduite telle, la seconde sans guillemets est calculée !

POINT-VIRGULE

> **Notez que le point-virgule sert de séparateur « officiel » dans un ordre PRINT. Il évite d'avoir à répéter deux fois PRINT, ici. L'affichage se fera à la suite, sans retour à la ligne.**

Essayons maintenant avec une soustraction, en exécution directe :

```
? 345-43
302
```

Curieux, ce zéro ...

La soustraction est classique, mais pas le zéro.

ZERO ET LETTRE O

En informatique, il ne faut pas confondre le chiffre zéro et la lettre O. Aussi écrit-on zéro avec une barre oblique. Ne vous trompez surtout pas en frappant O ou Ø !

MULTIPLICATION (*)

DIVISON (/)

On peut, maintenant expérimenter la multiplication et la division, mais il nous faut noter deux points importants.

Le symbole de la multiplication n'est pas le signe habituel × que l'on confond trop facilement avec la lettre x, mais l'astérisque (*). Le symbole de la division est la barre oblique (/).

Voici ce que donnerait une opération compliquée :

Multiplié par ...

... Et divisé par

```
? 2345*67/5
31423
```

Expérimentons maintenant une opération donnant à la fois un résultat *négatif* et une valeur décimale :

```
? -345/7
-49.2857143
```

Le signe «moins» a bien été placé

La réponse est : – 49,2857143

POINT DECIMAL

VIRGULE DECIMALE

Notez la précision de la réponse, fournie avec 7 décimales. Nous devons aussi attirer votre attention sur le fait que le Basic, d'origine américaine, utilise *un point décimal* au lieu de notre *virgule habituelle.*

N'oubliez jamais que le point décimal remplace la virgule décimale. Nous écrirons désormais tous nos nombres décimaux avec un point.

Maintenant que vous connaissez les règles, vous pouvez expérimenter toutes sortes d'opérations compliquées. Ne vous en privez pas. Vous apprécierez ce mode direct « calculette » qui, en fait, se révèle d'une puissance remarquable.. Vous pourrez aussi introduire des parenthèses à volonté dans vos expressions.

ELEVATION A PUISSANCE

Puisque nous sommes en pleine arithmétique, on va vous indiquer aussi comment élever un nombre à une puissance, et par exemple, calculer 3^2.

Le symbole de l'élévation à une puissance est une flèche verticale, dont vous trouverez la tête sur la touche du 6. Vous l'obtiendrez donc avec SHIFT.

```
? 3↑2
9
```

3 elevé au carré

A l'inverse, l'extraction d'une racine carrée s'écrit SQR. Ainsi :

RACINE CARREE

SQR

```
? SQR (9)
3
```

« Racine carrée de ... »

Placez obligatoirement la valeur (entre parenthèses)

Si vous disposez d'un Oric – 1, exercez-vous sur diverses opérations !

3. Point-virgule et virgule

On a déjà employé le point-virgule ; recommençons avec les dix premiers chiffres :

```
? Ø;1;2;3;4;5;6;7;8;9
Ø 1 2 3 4 5 6 7 8 9
```

Chaque chiffre est reproduit, séparé par un espace du précédent

VIRGULE DANS UN PRINT

Recommencez l'expérience, maintenant, en plaçant une *virgule* et non un point-virgule ; cela donnera :

```
? Ø,1,2,3,4,5,6,7,8,9
Ø      1      2      3      4      5      6      7
   8      9
Ready
```

4 espaces entre deux chiffres

Les chiffres ont été espacés de 4 espaces (« blancs »), même lorsqu'il y a eu retour à la ligne.

La virgule provoquera systématiquement l'introduction de 4 espaces, quelle que soit la longueur du nombre affiché ou du message, car on peut séparer par des virgules des messages alphabétiques encadrés par des guillemets.

EFFACEMENT DE L'ECRAN
CTRL/L

Votre écran est peut-être bien rempli, maintenant, Il a même probablement commencé à se « dérouler ». Voulez-vous l'effacer totalement ? Frappez CTRL/L : maintenez CTRL enfoncée et tapez L. Tout s'efface, le curseur se positionnant en haut et à gauche :

Il n'est pas nécessaire, ici, de faire RETURN. Puisque nous avons expédié le curseur à l'angle supérieur de l'écran, profitons-en pour expérimenter les touches de *déplacement du curseur,* ces 4 touches marquées des flèches ↓ ↑ → et ← de part et d'autre de la barre d'espacement. On verra leur utilité ensuite (pour dessiner, par exemple). Notez que quatre combinaisons, avec CTRL ont le même effet (vérifiez-le sur votre Oric) :

CTRL/H
CTRL/I
CTRL/J
CTRL/K

- CTRL/H déplace le curseur d'un pas vers la gauche ;
- CTRL/I en fait autant, mais vers la droite ;
- CTRL/J va vers le bas ;
- CTRL/K va vers le haut.

CTRL/M

Nous ne vous l'indiquons que pour mémoire : les touches fléchées sont plus pratiques à l'usage. Il en irait d'ailleurs de même avec RETURN, qui peut être remplacé par CTRL/M : pourquoi faire simple quand on peut faire compliqué ?

4. Variables et constantes

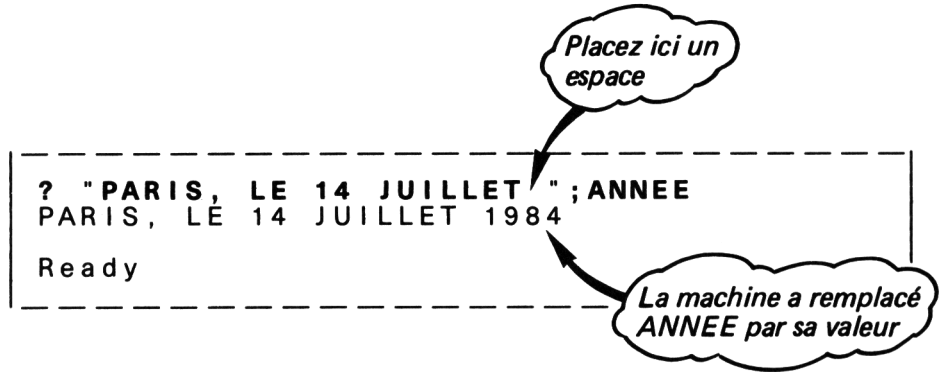
Lorsqu'on jongle avec des chiffres ou des lettres, il est parfois très agréable de leur décerner un nom. Si par exemple, nous pensons à 1984, nous pourrions appeler cela tout simplement *ANNEE*. Prévenons la machine que, désormais, ANNEE va correspondre à 1984. Frappez simplement :

```
Ready
ANNEE=1984

Ready
```

Maintenant, vous pouvez demander à l'ordinateur de quelle année il s'agit en faisant PRINT ANNEE, ce qui donne :

```
? ANNEE
1984

Ready
```

Dans un ordre complet, on pourrait même écrire en utilisant notre fameux point-virgule, pour placer ANNEE à la suite :



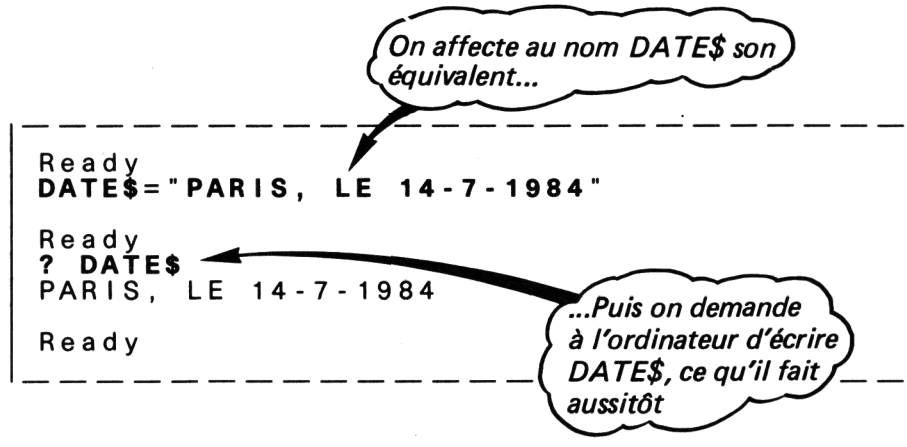
L'en-tête complète « Paris, le 14 juillet 1984 » peut, elle aussi, se voir remplacer par un nom ; appelons-le DATE.

CHAINES DE CARACTERES

SYMBOLE $

Il faut indiquer au Basic que DATE contient non seulement une suite de chiffres, mais aussi des lettres et même des symboles divers. C'est ce qu'on appelle une « chaîne de caractères » ; on la placera d'ailleurs entre guillemets. Pour cela, on va faire suivre son nom du symbole « dollars » ($).

Le symbole $ précise qu'il s'agit d'une *chaîne de caractères,* une série de caractères absolument quelconques et mélangés. On écrira donc DATE$, sans espace. Voici ce que cela donnerait :



Cette procédure est importante à bien comprendre : à un *nom,* on fait correspondre un *nombre* ou une *chaîne de caractères* (mais dans ce cas, le nom est immédiatement suivi du symbole *dollars* et la chaîne est placée « entre guillemets »). Recommençons, simplement avec des chiffres :

```
N=3

Ready
? N
3

Ready
? X
Ø

Ready
```

On affecte 3 à N ...

...Puis on demande N : c'est bien 3

Là on demande X, mais comme on ne lui a attribué aucune valeur au préalable, l'ordinateur décrète qu'il vaut zéro!

Sans affectation préalable, la valeur d'un nombre est zéro.

CONSTANTE

VARIABLE

En informatique, le « nom » est appelé une « variable » et ce qui lui correspond est une « constante ».

CONSTANTE NUMERIQUE

CONSTANTE CHAINE

VARIABLE NUMERIQUE

VARIABLE CHAINE

Ainsi, N est une *variable numérique* et 3, une *constante numérique.* On peut passer à une *variable chaîne* en ajoutant le suffixe $ à N ; dans l'exemple suivant, la *constante chaîne est BONJOUR :*

```
N$ = " BONJOUR "

Ready
? N$
BONJOUR

Ready
```

Notez que N$ est un nom différent de N tout court, pour le Basic. N vaut toujours 3, comme on l'a défini ci-dessus.

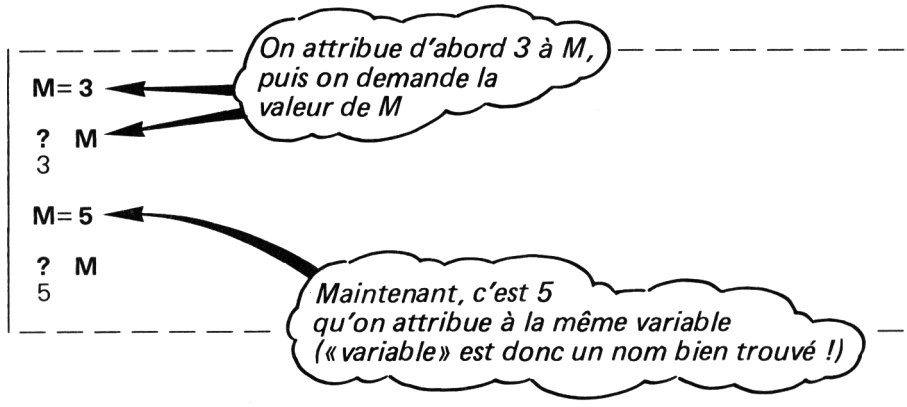
AFFECTATION

SYMBOLE D'AFFECTATION (=)

Il nous faut maintenant insister sur ce qu'on a appelé l'*affectation,* l'attribution d'une *constante* à une *variable.*

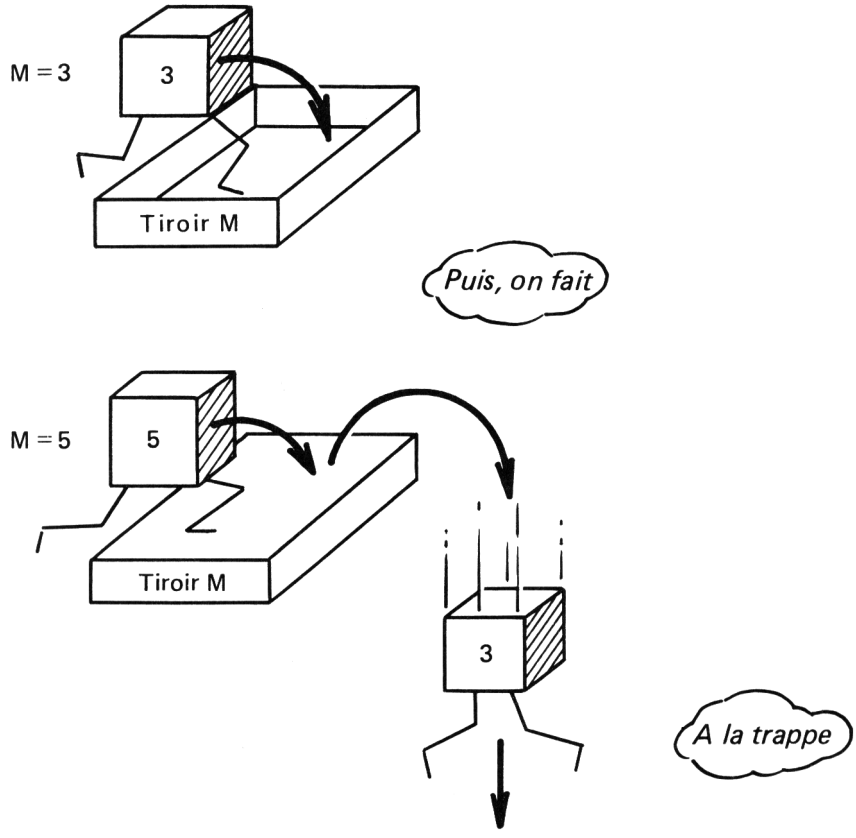
L'affectation est faite à l'aide du symbole égal (=). Mais attention : il ne s'agit plus du tout du symbole mathémathique « égal » et sa signification est ici tout à fait différente.

On a adopté ce symbole = car il a le mérite d'exister sur tous les claviers. Créer un nouveau symbole aurait posé des problèmes de clavier. Voici quelques exercices qui vont vous faire comprendre ce que signifie l'affectation (nous avons seulement supprimer les *Ready* entre les différents ordres que vous frappez et/ou les réponses de la machine) :

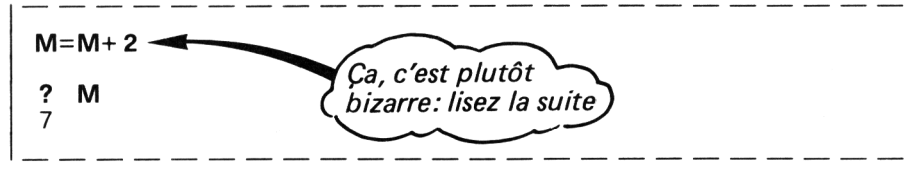


Une *variable*, c'est un nom, immuable : vous pouvez lui attribuer des valeurs successives mais attention : à un instant donné, *la variable ne peut se voir affecter qu'une seule constante.* Si vous lui affectez une nouvelle constante, la précédente est perdue ; on peut dessiner ainsi ce phénomène, en assimilant une *variable* à un *tiroir,* qui peut contenir des *constantes* successives :

Ordre :



Le nom M, représenté comme une boîte, correspond très exactement à une cellule (ou un groupe de cellules) de la mémoire centrale, qui ne peut contenir *qu'une seule valeur* à un moment donné. C'est pourquoi une nouvelle valeur chasse l'ancienne. Appliquons ce principe à quelque chose d'un peu plus fort, sachant que l'ancienne valeur de M est 5 :



Rappelez-vous : le signe = n'est plus une égalité mathématique, mais un symbole d'affectation.

Interprétez la première ligne ci-dessus par « *M se voit affecter l'ancienne valeur de M, plus 2* ». Ce qui fait bien 7.

NOMS DE VARIABLES

Quels noms pouvez-vous attribuer à vos variables ? Toutes sortes de noms, mais attention :

Votre Basic ne retiendra que les deux premiers caractères du nom attribué à la variable.

Ainsi, pour lui, AN et ANNEE sont le même nom, puisqu'il n'aura retenu que les deux premières lettres de ANNEE. Par conséquent, usez des noms dépassant deux caractères avec modération et rigueur ; dans ce cas, choisissez des noms expressifs, évocateurs : AGE pour l'*âge,* par exemple, et non XYZ.

Le nom de variable doit obligatoirement commencer par une lettre de l'alphabet, mais peut comprendre ensuite des chiffres.

Ainsi, M1 est autorisé, mais pas 1M . Nous vous conseillons d'ailleurs, très généralement :

- D'utiliser une unique lettre de l'alphabet comme nom de variable.
- Ou alors, deux lettres.
- Ou encore, une lettre et un chiffre.

Vous ajouterez au bout le symbole $ s'il s'agit d'une chaîne de caractères (nous allons y revenir).
Vous n'en êtes cependant encore pas quitte avec ces noms. En effet, toute une liste de mots est frappée d'interdit :

MOTS RESERVES

- D'abord, tous les mots que le Basic utilise pour ses propres besoins. Par exemple, PRINT ne peut pas servir comme nom pour une variable. La liste complète de ces mots interdits dont le Basic se réserve l'usage et qu'on appelle, pour cela, *mots réservés,* est donnée en fin d'ouvrage (c'est la liste des ordres Basic).

- Ensuite, des mots *incluants* ces mots réservés. Par exemple, ON est un mot réservé ; donc, vous ne pouvez pas employer ON, non plus que DON, ou MAISON, ou SONGE...

Si vous tentiez de le faire, le Basic refuserait votre ordre en affichant un « ?SYNTAX ERROR ». Ne vous entêtez pas (il est plus têtu que vous !), changez de nom ! Notez aussi que tous les noms de variables doivent s'écrire en majuscules.

CHAINES DE CARACTERES

5. Chaînes de caractères

Revenons aux chaînes de caractères, qu'on a définies comme étant une suite de caractères alphanumériques, symboles, espaces, etc. Voici quelques chaînes de caractères : « *Belle Marquise, vos beaux yeux me font d'amour mourir !* », ou « *Paris, le 14 juillet 1789* » ; *ou bien* « *Ah ?* » ; un espace peut constituer une chaîne : " ", et on peut même ne rien mettre dedans : ce sera une *chaîne vide,* qui s'écrit " ".

CHAINE VIDE

Un nombre pur devient une chaîne dès lors que vous le mettez entre guillemets ; ainsi, 1789 est une constante numérique mais pour le Basic, « 1789 » avec des guillemets devient une constante chaîne de caractères. Son nom de variable devra être suivi du symbole $.
Utilisons maintenant 3 chaînes ; observez ce qui va se produire. On a volontairement écrit des chaînes en majuscules et minuscules ; exercez-vous à taper ces deux sortes de caractères.

Rappelez-vous : à la mise sous tension, l'Oric ne fournit que des majuscules. Pour passer au mode « machine à écrire » avec majuscules et minuscules, faites CTRL/T.

Vous aurez directement les minuscules ; les majuscules seront obtenues avec SHIFT maintenue enfoncée. Pour revenir au mode « tout en majuscules », refaites CTRL/T.

Attention, tous les ordres Basic et les noms de variables doivent être écrits en majuscules.

Exercez-vous donc à utiliser aussi les minuscules !

```
A$="Bonjour"
B$="  "
C$="Madame"
? A$+B$+C$
Bonjour Madame
```

Cette chaîne ne comporte qu'un unique espace

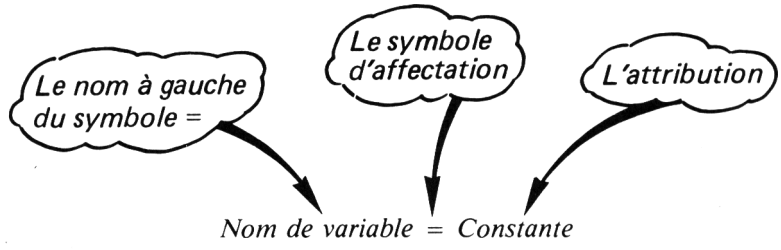
La dernière ligne peut surprendre : on a mit bout à bout les trois chaînes, ce qu'on appelle « concaténer ».

CONCATENATION DE CHAINES

SYMBOLE + DE CONCATENATION

Le symbole + n'est plus, ici, le symbole de l'addition mais celui de la concaténation.

Donc, A$ + B$ + C$ signifie tout simplement concaténer les trois chaînes.
Généralisons, maintenant, l'ordre d'affectation. C'est le suivant :



SONORISATION DES TOUCHES

CTRL/F

Avant de terminer ce chapitre, ajoutons encore quelques petites choses, relatives au clavier. Chaque frappe est *sonorisée,* ce que l'auteur apprécie (on « entend » qu'on a frappé un caractère). Si, toutefois, vous vouliez supprimer ce son, faites CTRL/F : les frappes ne seront plus sonorisées. Pour revenir à l'état antérieur, refrappez CTRL/F.

EFFACEMENT D'UNE LIGNE
CTRL/N

A propos de CTRL, sachez aussi que CTRL/N efface une ligne sur l'écran, celle qui est marquée par le curseur. Vérifiez-le, en emmenant le curseur sur des lignes existantes, puis en faisant CTRL/N. On usera très modérément de cet ordre.

QUESTIONS SUR LE CHAPITRE III

C'est facile ; cochez la bonne réponse !

1. L'ordre PRINT signifie :
   - ☐ Afficher sur l'écran
   - ☐ Exécuter une opération

Que répondra la machine aux ordres suivants :

2. PRINT "3+2"
   - ☐ 3+2
   - ☐ 5

3. PRINT 3+2
   - ☐ 3+2
   - ☐ 5

4. PRINT "3+2
   - ☐ ?SYNTAX ERROR
   - ☐ 3+2

5. ? 3+2
   - ☐ ?SYNTAX ERROR
   - ☐ 5

6. ? 1Ø+1O
   - ☐ 2Ø
   - ☐ 11 Ø

7. ? 7×3
   - ☐ 21
   - ☐ 7 Ø

8. ? 3↑3
   - ☐ ?SYNTAX ERROR
   - ☐ 27

9. ? "9/3=";4+2
   - ☐ 3=4+2
   - ☐ 9/3=6

Bon. Et maintenant :

1Ø. A quoi sert la virgule dans un ordre PRINT.
   - ☐ A séparer des constantes
   - ☐ A introduire 4 espaces dans leur affichage

11. A$ est un nom de variable :
   - ☐ Numérique
   - ☐ Chaîne

12. A-t-on le droit d'écrire successivement.
   T=3
   T=3*2
   T=T/6

   Et que donnera l'ordre d'afficher T ?

Réponses page 30

CHAPITRE IV

LE MODE PROGRAMMÉ

Le chapitre précédent ne constituait qu'une petite mise en condition : on va, maintenant, passer à la création de programmes réels. Pour cela, on va rédiger des listes d'instructions qui ne seront, cette fois, exécutées en salve que sur un ordre donné, et non plus une par une, au fur et à mesure de leur frappe. Ce mode s'appelle le « mode programmé », par opposition au « mode direct ».

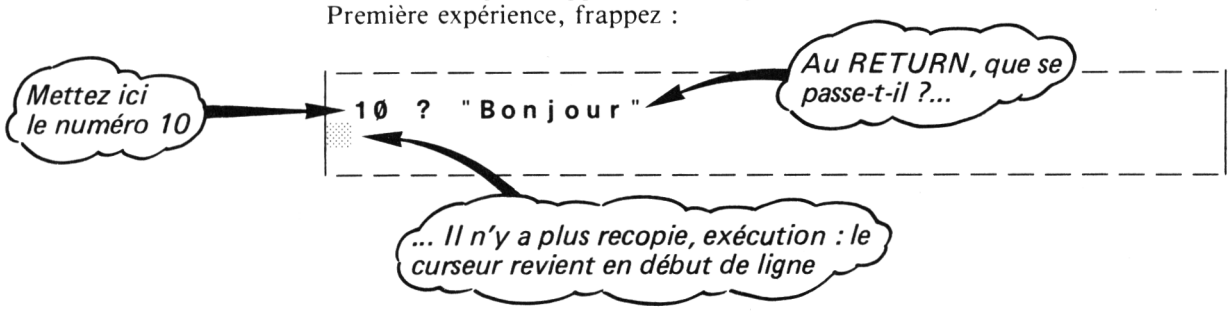
Principaux thèmes étudiés

Numérotation des lignes
RUN
Adjonction d'une ligne
LIST
Mode programmé
Remplacement d'une ligne
Suppression d'une ligne
Messages d'erreur
Effacement de la mémoire
NEW
Programme de conversion
INPUT
GOTO
Organigramme
REDO FROM START
CTRL/C
Branchement inconditionnel
Effacement programmé de l'écran
CLS
Pour sauter une ligne
CONT
Code ASCII
CHR$
Listage partiel
Modification d'une ligne
CTRL/A
Temporisation
WAIT
EDIT

1. Numérotation des lignes

NUMEROTATION DES LIGNES

Dans ce mode *direct* étudié dans le chapitre précédent, un ordre frappé au clavier était immédiatement exécuté, dès la frappe sur la touche RETURN. Or, un programme informatique peut se composer de plusieurs ordres successifs (jusqu'à des milliers...) : par conséquent, il faut pouvoir les ranger en mémoire *sans* les exécuter au fur et à mesure ; on passera à cette exécution au seul moment opportun. C'est un tel mode qu'on appelle le *mode programmé,* ou encore le *mode programme.* Première expérience, frappez :



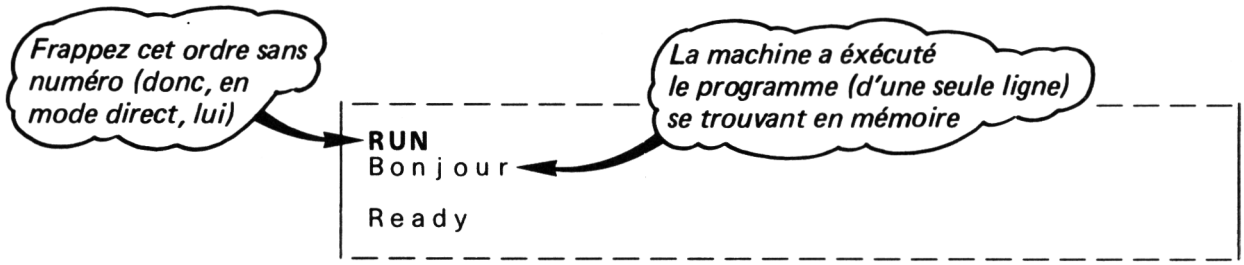
La machine n'a pas exécuté le PRINT « Bonjour » et le curseur est revenu à la ligne, attendant une nouvelle instruction.

Le seul fait de faire précéder l'instruction par un numéro d'ordre fait entrer la machine en « mode programmé ».

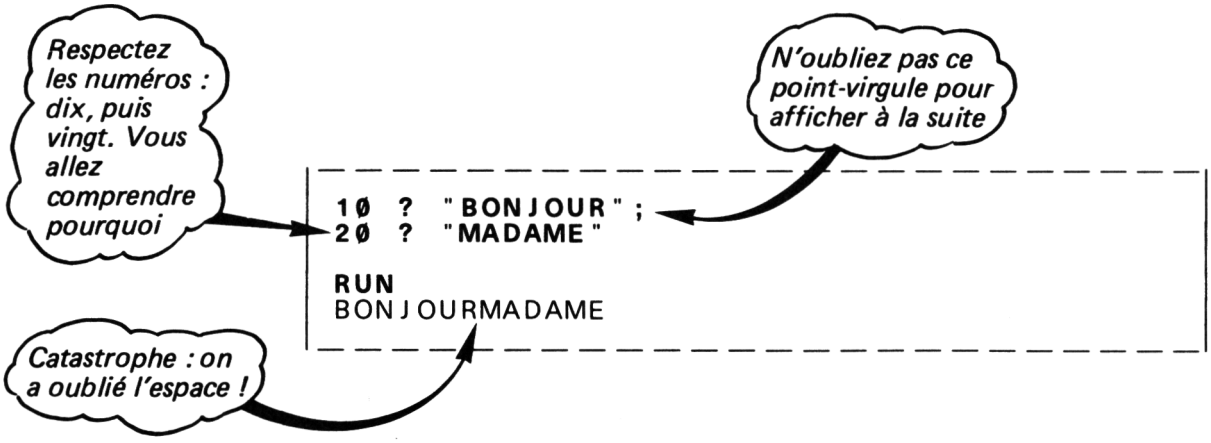
Le numéro d'ordre, ou numéro de ligne d'instruction, peut être quelconque. Mettez 1, ou 2, ou 3, ou 78, ou 357,... peu importe : l'essentiel, c'est la présence d'un numéro.

RUN

Qu'est devenue l'instruction numérotée 1Ø, alors ? *Elle a été rangée en mémoire par le Basic, dès la frappe de RETURN.* En voulez-vous la preuve ? On va lancer son exécution : pour cela, il faut utiliser le mot magique RUN (quelque chose comme : *Marche !*). Essayez tout d'abord de frapper ces trois lettres, suivies d'un RETURN :

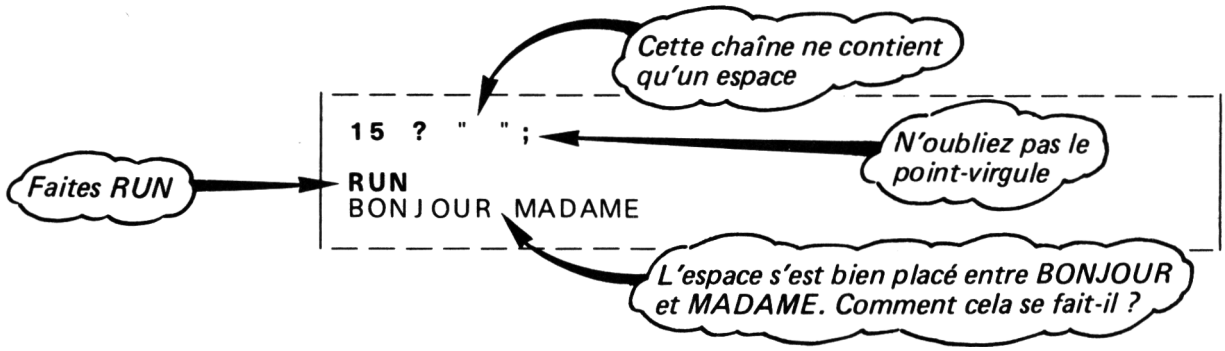


Vous pouvez désormais relancer ce programme autant de fois que vous le voudrez, sans refrapper l'instruction, simplement avec RUN. Vérifiez-le : vous obtiendrez à chaque fois le même résultat. Cette ligne 1Ø constitue un *programme* minimal : passons à un programme de longueur double. Frappez maintenant une nouvelle ligne 1Ø, terminée par un point-virgule, puis la ligne 2Ø suivante, et faites RUN :



ADJONCTION D'UNE LIGNE

L'ordre RUN — donné en *mode direct,* lui, remarquons-le à nouveau — provoque l'affichage de *BONJOURMADAME* car on a oublié un espace. Qu'à cela ne tienne ; créons-en un et pour cela, imaginons une instruction, de numéro 15, qui devrait s'intercaler entre 1Ø et 2Ø. Quel que soit la ligne au début de laquelle votre curseur se trouve, frappez :



Avez-vous bien saisi, maintenant, le rôle du point-virgule dans un ordre PRINT ? Il supprime le retour à la ligne ; l'affichage suivant se fera alors sur la même ligne.

Pour comprendre ce qui s'est passé avec les lignes du programme, demandez au Basic de vous fournir le *listage* complet du programme qu'il a actuellement en mémoire, donc des trois lignes d'instructions que vous avez frappées.

LIST

Demandez le listage se dit LIST. Frappez ces quatre lettres, puis faites RETURN. C'est donc un ordre en mode direct.

Voici ce que cela donnera :

N'oubliez pas RETURN, ici

```
LIST
10 PRINT "BONJOUR";
15 PRINT " ";
20 PRINT "MADAME"

Ready
```

Remarquez que le Basic a remplacé les points d'interrogation par le mot qu'ils représentent, PRINT. Il vous affiche vos trois lignes et a *introduit la ligne 15 entre la 10 et la 20.*

Vous comprenez, maintenant, pourquoi on a numéroté les lignes 10, puis 20 (on aurait continué par 30, puis 40, etc.) ? Lorsqu'on programme, on risque toujours d'oublier une instruction : elle sera facile à introduire si l'on s'est réservé des places dans la numérotation.

Tenez compte d'un autre phénomène important : le Basic ne s'intéresse qu'à l'ordre croissant des numéros, et non à leur valeur « absolue ». Il rangera les instructions en mémoire selon leur ordre croissant et les exécutera dans le même ordre, quel que soit le « pas ». Ainsi, au lieu d'écrire 15 pour l'instruction manquante, on aurait pu mettre 11, ou 12, ou 13, ou 14, ou... 19 : elle se serait placée entre 10 et 20 de la même façon.

Prenez donc, vous aussi, l'habitude de numéroter les instructions par « pas » de 10.

MODE PROGRAMME

Accessoirement, remarquez autre chose. On vient de voir que le fait de commencer une ligne par un numéro provoque le passage en *mode programmé ;* or, rappelez-vous ce qu'on a dit dans le chapitre precédent, à propos des noms de variables : ils ne doivent *pas* commencer par un chiffre. Dans ce cas, en effet, le Basic imaginerait que vous avez voulu numéroter une ligne. Si vous écriviez :

C'est un nom de variable numérique interdit

```
5R=8
```

puis faisiez RETURN, le Basic penserait qu'il s'agit de la ligne 5 avec l'affectation R = 8. Il introduirait donc cette ligne, comme le prouve le listage :

```
LIST

5  R=8
10  PRINT  "BONJOUR";
15  PRINT  "  ";
20  PRINT  "MADAME"
```

Le nom (interdit) de variable 5 R a été interprêté comme la ligne 5, avec la variable R

Laissons cette ligne 5, pour l'instant, car elle n'interviendra pas dans l'exécution du programme (qui ne demande pas l'affichage de R).

2. Les corrections

REMPLACEMENT D'UNE LIGNE

Supposons qu'au lieu d'écrire *Bonjour Madame,* on veuille mettre *Bonjour Monsieur :* il faut *modifier* la ligne 20, en créer une autre plus exactement. Frappez-là ainsi :

```
20  ?  "MONSIEUR"
```

Puis, faites RETURN et LIST, afin de vérifier ce qui s'est passé. Vous lirez :

```
LIST

5  R=8
10  PRINT  "BONJOUR";
15  PRINT  "  ";
20  PRINT  "MONSIEUR"
```

La nouvelle ligne 20 *a pris la place* de l'ancienne, qui est perdue.

Pour remplacer une ligne d'instruction par une autre, il suffit de frapper la nouvelle instruction avec le même numéro de ligne.

SUPPRESSION D'UNE LIGNE

En poussant ce principe à son extrême, on peut se demander ce qui se passerait si l'on frappait un numéro de ligne existant, puis *rien* après, sauf le RETURN ? Et bien, la nouvelle ligne *vide* prendrait la place de l'ancienne, l'effacerait. Vérifions-le avec la ligne 5, inutile :

```
5
LIST

10  PRINT  "BONJOUR";
15  PRINT  "  ";
20  PRINT  "MONSIEUR"
```

Faites ici RETURN

L'ancienne ligne 5 a bel et bien été supprimée

Pour supprimer une ligne de programme, il suffit de frapper son numéro puis de faire RETURN.

Rappelez-vous aussi que, pour supprimer une ligne en cours de frappe, donc avant d'avoir fait RETURN, on peut frapper CTRL/X qui l'annule. Ou alors, faire des DEL successifs.

CTRL/X place une barre oblique inversée à la fin de la ligne en cours, l'annule, et ramène le curseur au début de la ligne précédente.

Enfin, notez encore un point important, afin de ne pas être surpris par la suite. Si vous frappez une instruction comportant une erreur, le Basic pourra :

MESSAGES D'ERREUR

• Soit la détecter aussitôt, dès la validation par RETURN et vous le signaler en refusant l'ordre (par exemple, en affichant « ?SYNTAX ERROR »).

• Soit ne pas la détecter au RETURN, donc l'accepter, mais s'apercevoir de l'erreur au RUN. Dans ce cas, il vous affichera l'un de ces nombreux messages d'erreur dont il a le secret.

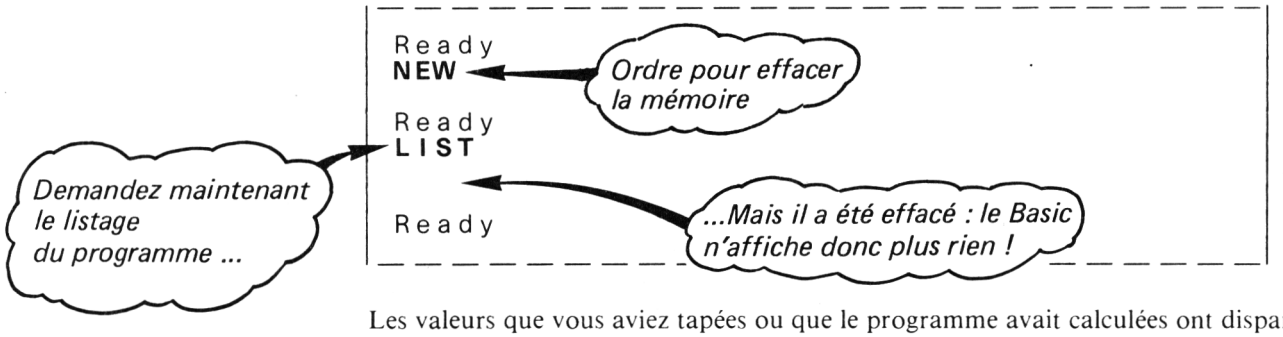
Si, donc, après un RUN l'ordinateur vous indique une erreur (en précisant son numéro de ligne), vérifiez attentivement la ligne indiquée pour la découvrir. Même si cette erreur ne vous saute pas immédiatement aux yeux, restez persuadé que c'est le Basic qui a raison et qu'il y en a bien une. A ce propos, méfiez-vous des noms de variables comportant un mot réservé : ce n'est pas toujours évident.

3. Effacements de la mémoire et de l'écran

EFFACEMENT DE LA MEMOIRE

NEW

On a déjà introduit plusieurs ordres en mémoire ; comment les effacer en bloc, toutes les lignes d'un coup ? Une méthode radicale consiste à arrêter un instant l'Oric – 1, puis à le remettre en service ; ses mémoires vives étant *volatiles,* elles vont perdre toutes leurs informations.
Une méthode plus élégante consiste à signifier à l'ordinateur qu'on veut libérer les mémoires pour créer un nouveau programme. C'est l'ordre NEW, pour *nouveau ;* « entrez-le » en mode direct (le mot *entrer* fait partie du jargon de l'informatique ; il signifie : *frapper au clavier puis valider avec RETURN).* Si, aussitôt après, vous demandez un LIST, vous n'obtiendrez plus rien :



Les valeurs que vous aviez tapées ou que le programme avait calculées ont disparu aussi, mais l'écran reste bien rempli ; profitons-en pour l'effacer. Vous souvenez-vous comment ? Oui, bravo : avec CTRL/L. Maintenez CTRL enfoncée et frappez L : tout s'efface et le curseur revient à l'angle supérieur gauche.

4. Quelle est la vitesse du bateau ?

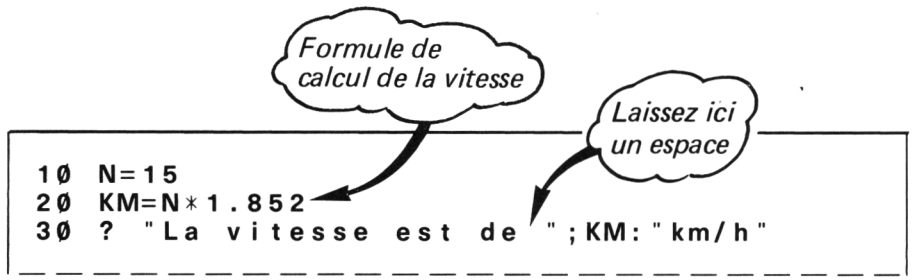
On va profiter de ce bel écran pour rédiger un programme de caractère utilitaire : à

PROGRAMME DE CONVERSION

quoi correspond, en kilomètres par heure, la vitesse d'un bateau donnée en *nœuds ?* Dire qu'un bateau « file 15 nœuds » signifie qu'il se déplace à la vitesse de 15 *milles marins* à l'heure, un *mille marin* valant 1 852 mètres (toutes choses que vous saviez). Le programme suivant va convertir les nœuds en kilomètres-heure ; il consiste à multiplier par 1,852 la vitesse en nœuds.

Rappelez-vous qu'en Basic, la virgule décimale est remplacée par un point décimal et que le symbole de la multiplication est l'astérisque (*).

Entrez en machine le programme suivant, où N est le nom de variable de la vitesse, en nœuds, et KM le nom de variable de la vitesse en km/h :



En frappant ce programme, rappelez-vous que le texte des messages entre guillemets peut être en minuscules, mais jamais les noms des variables ni d'ailleurs les ordres, tels que RUN, ou LIST, obligatoirement en majuscules.

A la ligne 10, on attribue la constante 15 à la variable N (nœuds) ; en 20, on calcule la vitesse en kilomètres-heure, mais on n'affiche toujours rien. Ce n'est qu'en 30 qu'on commande l'affichage PRINT (avec ?) d'une formule complexe : tout ce qui est placé « entre guillemets » va être reproduit tel, mais au centre, KM sera remplacé par sa valeur.

Lancez ce programme avec RUN et vous obtiendrez en-dessous :

```
RUN
La vitesse est de 27.78 km/h

Ready
```

En effet, 15 nœuds correspondent bien à 27,78 km/h, puisque c'est l'ordinateur qui le dit. Essayez d'autres vitesses en nœuds, en refrappant la ligne 10 et en mettant d'autres valeurs à la place de 15. Puis, quand vous en aurez assez de refrapper cette ligne, nous vous montrerons une façon plus intéressante de procéder.

5. L'ordre d'entrée INPUT

INPUT

Cette fois, on va confier au programme le soin de vous interroger sur la vitesse en nœuds choisie. Pour cela, on remplace la ligne 10 par une nouvelle ligne comportant un ordre de « demande d'entrée d'une nouvelle donnée », ce qui se dit INPUT *(entrez)* en Basic.
Vous vous souvenez que, pour remplacer une ligne, il suffit de refrapper la nouvelle avec le même numéro. Faites donc, en frappant les cinq lettres de INPUT :

Vérifiez, avec LIST, que la substitution s'est bien faite :

```
LIST

10  INPUT  N
20  KM=N*1.852
30  PRINT  "La vitesse est de ";KM;"km/h
"
```

Tout va bien

Parce que le Basic écrit le mot PRINT en toutes lettres dans le listage, il provoque un malencontreux retour à la ligne. Tant pis : il faut s'y faire.

Effacez donc à nouveau l'écran avec CTRL/L, puis lancez un RUN et observez ce qui s'affiche : un simple point d'interrogation au début de la ligne ! Cela donne :

```
RUN
?
```

Le curseur est ici

La machine a posé ce point d'interrogation

En effet, lorsque le Basic rencontre un ordre INPUT (ce qu'il vient de faire), il affiche un ? et stoppe toute activité, attendant que vous frappiez une donnée au clavier que vous validerez avec un RETURN. Dès lors, il poursuit l'exécution du programme.

Attention : ne confondez pas le ? affiché par le Basic qui vient de rencontrer un INPUT avec le ? que vous placez pour remplacer PRINT. C'est bien le même symbole, mais pour deux choses différentes.

En réponse à la demande du programme, frappez par exemple 9, pour voir, puis faites RETURN ; l'affichage complet sur l'écran sera :

```
RUN
? 9
La vitesse est de 16.668 km/h

Ready
```

Vous avez frappé ce 9, puis fait RETURN

Vous pouvez frapper à nouveau sur RUN et introduire d'autres valeurs, l'essentiel étant que vous compreniez parfaitement le fonctionnement de INPUT.

6. Et si l'on recommençait ?

Si vous aviez de multiples conversions à exécuter, les frappes successives de RUN pour relancer le programme seraient fastidieuses. Dans ce cas, il est plus simple d'ajouter une nouvelle instruction, à la fin de ce programme, qui lui ordonne de recommencer en renvoyant à son début.

GOTO

Cette nouvelle instruction portera le numéro 4Ø. Elle ordonnera au programme d'aller sans discussion à l'instruction 1Ø, donc de revenir à son début. *Aller à* se dit *go to,* en anglais ; et s'écrit en un seul mot GOTO. Entrez :

```
40 GOTO 10
```

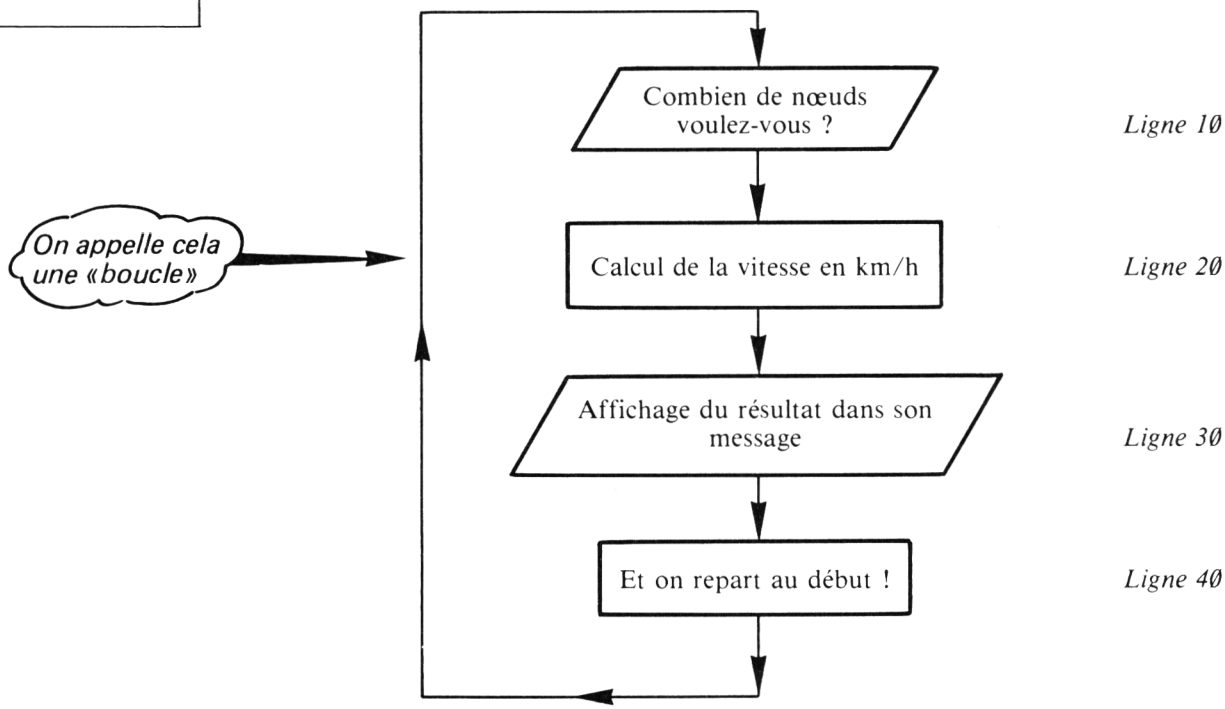
«Aller obligatoirement à 10»

Faites LIST, pour voir où vous en êtes :

```
LIST

10 INPUT N
20 KM=N*1.852
30 PRINT "La vitesse est de ";KM;" km/h
"
40 GOTO 10
```

ORGANIGRAMME

On peut représenter ce programme par un petit dessin qu'on appelle un *organigramme*. Le voici :



Organigramme du programme de conversion

Au passage, notez que l'organigramme répond à quelques règles de dessin qui veulent que les ordres d'*entrée* (INPUT) ou de « sortie » vers l'écran (PRINT) soient inscrits dans des parallélogrammes, alors que les ordres d'exécution « internes » à la machine figurent dans des rectangles.
Toujours est-il que cet organigramme montre bien que, de la fin du programme, on repart à son début en une « boucle ». Faites RUN ; la machine va vous interroger une première fois : supposons que vous frappiez 9. Elle va afficher le résultat puis vous ré-interroger à nouveau : on suppose que vous frappez 11 ; puis ça continue, et vous frappez 7... Cela donnera sur l'écran :

```
RUN
? 9
La vitesse est de 16.668 km/h
? 11
La vitesse est de 20.372 km/h
? 7
La vitesse est de 12.964 km/h
?
```

La machine ne se lasse pas et en redemande ... On suppose que vous avez fini vos conversions

REDO FROM START

En effet, elle obéit aveuglément au programme et ne saura pas que vos calculs sont terminés. Si vous frappiez n'importe quoi d'autre que des chiffres, elle serait interloquée :

De guerre lasse, vous avez frappé ceci

```
? Mais j'ai fini !
?REDO FROM START
?
```

Traduction (approximative) de la réponse de la machine : « Je ne veux pas le savoir; recommencez votre entrée ! »

Elle attend, en effet, la frappe d'un nombre

CTRL/C

La formule « REDO FROM START » signifie, en effet : « Refaites à partir du début votre frappe ». Comment indiquer à cette obstinée que nous en avons terminé ? Simplement en frappant la combinaison CTRL/C, ce qui donnera, si l'on reprend la fin de l'écran précédent :

```
?REDO FROM START
?

Ready
```

Ici, vous avez frappé CTRL/C

La machine est. désormais, prête à accepter d'autres ordres, même un nouveau RUN pour relancer le même programme.
En général, lorsqu'on utilise une instruction INPUT, on s'arrange pour qu'un message explicatif précède le point d'interrogation sur l'écran. Vous obtiendrez ce résultat en créant une nouvelle ligne, 5 par exemple ; mais il faudra *aussi* que le GOTO de la ligne 40 renvoie en 5 et non plus en 10, ce qui va vous obliger à refrapper ces deux lignes :

```
5 ? "Donnez la vitesse en noeuds:"
40 GOTO 5
```

De ce fait, le programme complet, suivi de son lancement, donne :

```
LIST

5 PRINT "Donnez la vitesse en noeuds:"
10 INPUT N
20 KM=N*1.852
30 PRINT "La vitesse est de ";KM;"km/h
"
40 GOTO 5

RUN
Donnez la vitesse en noeuds :
? 6.78
La vitesse est de 12.55656 km/h
Donnez la vitesse en nœuds:
?
```

Etc, jusqu'à un CTRL/C

Faites un CTRL/C pour stopper ce programme.

BRANCHEMENT INCONDITIONNEL

Notez encore que l'ordre GOTO commande un branchement « inconditionnel », sans discussion, à la ligne indiquée. C'est pourquoi on l'appelle « branchement inconditionnel ».

7. Effacement programmé de l'écran

EFFACEMENT PROGRAMME DE L'ECRAN

CLS

On peut fignoler ce programme en commençant, dès le RUN, par effacer l'écran. Oui, mais comment introduire un CTRL/L dans le programme ? C'est parfaitement possible ; la preuve, on vous montrera comment plus loin. Mais la méthode la plus simple consiste à introduire l'ordre d'effacement de l'écran, ce qui se dit en anglais « *Clear screen* » *(effacer l'écran)* et s'écrit, en abrégé, CLS. Essayez-le d'abord en mode direct ; frappez CLS, puis RETURN, et vous obtiendrez le bel écran suivant :

```
Ready
```

Après CLS, en mode direct

Ayant vérifié que CLS fonctionne, vous pouvez l'introduire maintenant dans le programme, par exemple en ligne 3 :

```
3 CLS
```

Dès lors, et avec le RUN, *le programme commence par effacer l'écran ;* vous n'y trouverez plus que les lignes dont vous avez commandé l'affichage.
Un dernier défaut subsiste encore : les réponses ne sont pas séparées de nouvelles questions ; introduisons un saut de ligne, d'un calcul à l'autre, avec cette ordre bizarre à première vue :

```
35 ?
```

Qu'est-ce-que cela signifie ?

POUR SAUTER UNE LIGNE

Ici, on a frappé un numéro de ligne suivi par un ordre PRINT (? en abrégé), suivi lui-même par... rien du tout ! Dans ce cas, le Basic va lire ce PRINT et s'apercevra qu'il ne doit rien afficher ; par conséquent, il sautera une ligne. Revoyons le programme complet, d'abord :

```
LIST

3 CLS
5 PRINT "Donnez la vitesse en noeuds:"
10 INPUT N
20 KM=N*1.852
30 PRINT "La vitesse est de ";KM;"km/h
"
35 PRINT
40 GOTO 5

RUN
```

Pour sauter une ligne avant de recommencer

Avec le RUN puis le RETURN, vous verrez :

```
Donnez la vitesse en noeuds:
? 33
La vitesse est de 61.116 km/h

Donnez la vitesse en noeuds :
?
```

Ici, une ligne a été sautée

Etc, jusqu'au CTRL/C

Notez bien le rôle de ce PRINT suivi par rien du tout (si ce n'est RETURN) : il sert à sauter une ligne.

CONT

Deux PRINT successifs auraient servi à sauter deux lignes, par exemple.
Pour en terminer avec ce programme, vous faites un CTRL/C mais, pris de remords, vous voulez continuer vos conversions : que faire ? Frappez simplement CONT, pour continuer, et il reprendra au point exact où vous l'aviez lâché. La séquence serait la suivante :

```
Donnez la vitesse en noeuds:
? 5
La vitesse est de 9.26 km/h

Donnez la vitesse en noeuds:
?

Ready
CONT
?
```

Vous avez fait, ici, un CTRL/C ...

... Mais vous vous reprenez et voulez continuer : frappez ceci

Le programme reprend et attend une nouvelle valeur à convertir

Dès lors, vous pouvez effectivement continuer.

8. Le code ASCII

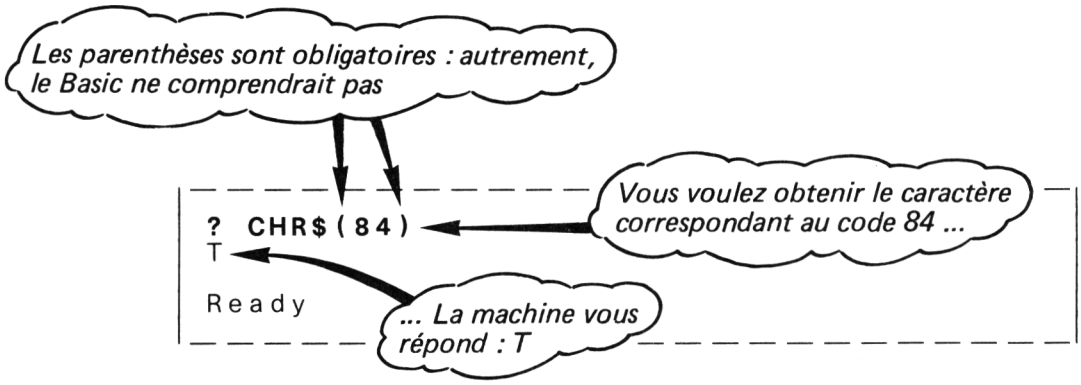
Sachez une chose importante : chaque caractère que vous pouvez frapper au clavier, qu'il soit simple ou combiné avec SHIFT ou CTRL, dispose d'un code numérique allant de Ø à 255.

CODE ASCII

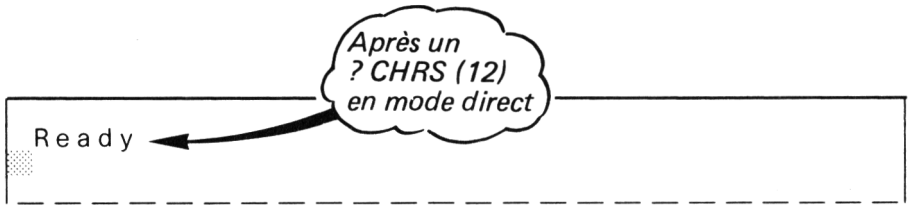
L'ensemble de ces codes est normalisé (plus ou moins...) et relève des normes ASCII, ce qui se prononce *Aski* et vient de « *American Standard Code for Information Interchange* ». Les normes strictes ASCII, allant de 32 à 127, sont données en fin de ce livre ; aux autres codes correspondent des ordres divers ou des caractères graphiques sur lesquels on reviendra.

CHR$

On peut demander au Basic un caractère quelconque en frappant son code. Sachant que *caractère* se dit « *character* » en anglais, on comprendra que l'ordre se code CHR ; en outre, il doit être suivi du fameux symbole *dollars* ($) puisqu'il s'agit de caractères précisément. On écrira donc PRINT CHR$ suivi du code du caractère voulu. Le T, par exemple, a pour code 84, ce que vous pouvez vérifier dans le tableau ASCII ; pour afficher un T, vous pourrez écrire (faites-le en mode direct) :



Tentez votre chance avec d'autres valeurs ; par exemple, le *t* minuscule se code 116 : il y a toujours une différence de 32 entre majuscules et minuscules.
Or, l'effacement qui se frappe avec CTRL/L, possède aussi son code : c'est le 12. Vous devinez alors que l'on puisse aussi commander l'effacement avec ? CHR$(12). Vérifiez-le en mode direct : l'effet sera semblable à celui de CLS, le mot « *Ready* » venant alors toutefois sur la 3e ligne et non sur la seconde :



Vous pouvez introduire cet ordre dans notre programme précédent, en 3, à la place de CLS :

Relancer le programme : vous n'y verrez guère de différence, et vous n'en verrez plus du tout si vous ajoutez un point-virgule au bout de la ligne 3, puisque grâce à lui vous supprimez un passage à la ligne !

9. L'édition

Vous l'apprendrez vite à vos dépens, la mise au point d'un programme fonctionnant comme on l'espérait impose bien des retouches et des corrections. Nous avons déjà passé en revue certaines méthodes, mais il vous faut d'emblée aller encore un peu plus loin.
Vous serez amené à vous servir fréquemment de ces diverses méthodes ; c'est pourquoi nous vous invitons à lire ce passage, à expérimenter ce que nous vous proposerons, puis à y revenir dès lors que le besoin s'en fera sentir.
Considérons l'ordre LIST. Vous pouvez très bien *lister* une seule ligne, par exemple la ligne 1Ø du programme précédent (nous supposons qu'il est toujours en mémoire) :

```
LIST 1Ø
1Ø INPUT N
Ready
```

Vous demandez la seule ligne 10 ...

...Qui apparait aussitôt

LISTAGE PARTIEL

Vous pourriez demander le listage du programme jusqu'à la ligne 2Ø incluse, par exemple, ce qui s'écrirait :

N'oubliez pas le tiret (signe «moins»)

```
LIST -2Ø
3 PRINT CHR$(12)
5 PRINT "Donnez la vitesse en noeuds:"
1Ø INPUT N
2Ø KM=N*1.852
Ready
```

Voici les 4 lignes, jusqu'à 20

Ou encore, vous pourriez souhaiter les lignes à partir de 3Ø incluse :

N'oubliez pas le tiret après le numéro de ligne, cette fois

```
LIST 3Ø-
3Ø PRINT "La vitesse est de ";KM"km/h
"
35 PRINT
4Ø GOTO 5
Ready
```

Voici le listage, de 30 à la fin du programme

Dernière formule, enfin, le listage à partir de 5, par exemple, jusqu'à 2Ø inclus ; voici :

```
LIST 5-20

5 PRINT "Donnez la vitesse en noeuds:"
10 INPUT N
20 KM=N*1.852

Ready
```

MODIFICATION D'UNE LIGNE

Supposons maintenant que vous vouliez modifier la ligne 40 et écrire à la place GOTO 3 (au fait, pouvez-vous deviner quel sera le résultat de cette modification ?). Vous pourriez refrapper totalement cette ligne (pourquoi pas ?). Mais il existe une méthode souvent préférable permettant d'apporter des modifications à une ligne sans avoir à la refrapper. Pour cela, il faut « éditer » cette ligne, ce qui signifie dans ce jargon très discutable de l'informatique : l'afficher (ou l'imprimer). C'est là où LIST intervient. Notre programme étant court, listez-le en entier. Puis :

1) — Servez-vous des touches fléchées (autour de la barre d'espacement) pour emmener le curseur tout au début de la ligne 40. Voici ce que cela donnera (on ne vous montre que cette seule ligne) :

Vous avez amené le curseur, ici,

```
40 GOTO 5
```

CTRL/A

2) — Servez-vous, maintenant, de la combinaison CTRL/A pour *déplacer le curseur vers la droite* (maintenez CTRL enfoncée et frappez plusieurs fois sur A). Avec CTRL/A, le curseur « relit » les caractères qu'il croise et, pour l'ordinateur, tout se passe comme si vous les refrappiez. Stoppez le curseur sur le 5.

3) — Maintenant, frappez votre correction, donc un 3 au lieu du 5. Vous pourriez de la même façon effacer des caractères avec DEL ; si d'autres caractères à conserver suivaient, il vous faudrait les « recopier » en promenant dessus le curseur, toujours avec CTRL/A.

Attention : seul CTRL/A relit et « recopie » en mémoire les caractères sur lesquels il passe. Cela, en mode « édition ».

4) — Faites RETURN : la nouvelle ligne est enregistrée.

5) — La correction est terminée, mais il reste encore une précaution à prendre : extrayez le curseur du corps du programme, emmenez-le au-dessous avec la touche marquée d'une flèche verticale (↓) car sinon, ce que vous frapperiez à nouveau viendrait se conjuguer avec les lignes existantes, donnant un résultat imprévisible. Ou alors, et plus simplement, effacez l'écran avec CTRL/L.
Vous pouvez maintenant vérifier l'état de la nouvelle ligne 40 modifiée, avec LIST, ou simplement LIST 40 ; vous obtiendrez :

```
LIST 40

40 GOTO 3

Ready
```

Que pensez-vous que cela donnera, avec un RUN ? A chaque nouvelle vitesse en nœuds frappée, l'ordinateur calcule la vitesse correspondante en kilomètres-heure, l'affiche, puis efface l'écran et repose la même question. Vérifiez-le avec un RUN. Hélas ! L'affichage du résultat est instantané et l'écran est effacé tellement vite qu'on n'a même pas le temps de le lire !

TEMPORISATION

WAIT

Aussi va-t-on prier l'ordinateur de bien vouloir attendre un instant, avant de ré-effacer l'écran. *Attendre* se dit WAIT, en anglais ; on va faire suivre WAIT d'une durée, sachant qu'une durée de 1 équivaut à 10 millièmes de secondes ; par conséquent, pour « temporiser » une seconde, on écrira WAIT 100 (car 100 fois 10 ms = 1 seconde). Introduisez cette instruction en remplacement de la ligne 35, qui n'a plus d'intérêt de toute façon :

```
35 WAIT 100
```

Pour mémoire, voici à quoi ressemble maintenant notre programme complet (vous l'obtenez, comme de coutume, avec un LIST) :

```
LIST

3 PRINT CHR$(12)
5 PRINT "Donnez la vitesse en noeuds:"
10 INPUT N
20 KM=N*1.852
30 PRINT "La vitesse est de ";KM;"km/h
"
35 WAIT 100
40 GOTO 3
```

Refaites un RUN : cette fois, l'affichage dure une seconde. Voulez-vous en modifier la durée ? Vous savez comment le faire sans refrapper la ligne, avec un LIST 35, les touches fléchées pour amener le curseur au début de la ligne, CTRL/A pour l'emmener sur le 1. Frappez alors un 2, par exemple, puis « recopiez » les deux zéros en frappant encore sur CTRL/A ; le curseur étant « sorti » du texte, faites RETURN, effacez l'écran, et lancez le programme avec RUN. La temporisation sera de 2 secondes.

EDIT

Voulez-vous la porter à 5 ? Il faut frapper 500, mais on va vous indiquer une dernière méthode. Faites apparaître la ligne 35 en utilisant, cette fois, un ordre qui porte bien son nom : EDIT. Ecrivez EDIT 35. Vous constaterez que le curseur reste alors sagement au début de la ligne 35 qui apparaît :

```
EDIT 35

35 WAIT 200
```

Vous pouvez emmener alors le curseur avec CTRL/A sur le 2 afin de le modifier en 5, tout comme on l'a fait précédemment, refaire CTRL/A pour « sortir » le curseur, faire RETURN : la modification est terminée et la temporisation a été portée à cinq secondes. Vérifiez-le avec un RUN.

Avec de la pratique, vous apprendrez à combiner ces diverses possibilités (nous ne vous en avons d'ailleurs pas tout dit ici), et vous acquerrez les automatismes qui vous rendront les modifications et corrections simples. Afin d'acquérir cette indispensable pratique, nous allons maintenant vous proposer à la fin de chaque chapitre, des programmes modèles, allant jusqu'aux jeux. N'abordez toutefois ces programmes qu'après avoir répondu *avec succès* aux exercices de la page 60.

Entrez les programmes dans votre Oric un seul à la fois..., ils pourraient se gêner, et essayez-les. Les exemples d'utilisation sont là pour vous guider. Grâce aux remarques décortiquez le programme et changez une virgule, remodelez à votre guise.
C'est l'occasion de mettre en pratique les modifications grâce à EDIT et CTRL/A dont on vient de vous parler.

Vous retrouverez les mots

INPUT PRINT WAIT GOTO

et vous consulterez une portion de la table de codification des caractères grâce à la fonction CHR$ et son complément ASC.

QUESTIONS SUR LE CHAPITRE IV

1. Y a-t-il une erreur dans la séquence suivante ?

```
1 ? "Bonjour,"
2 ? "chers amis."
999 ? "A bientôt."
```

2. Pour lancer l'exécution d'un programme, l'ordre est :
   - ☐ RUN
   - ☐ GO

3. Un point-virgule placé dans un ordre PRINT :
   - ☐ Commande un retour à la ligne
   - ☐ Supprime le retour à la ligne

4. Peut-on ré-afficher un programme se trouvant en mémoire ?
   - ☐ Oui, avec l'ordre LIST
   - ☐ Non, il faut le refrapper

5. Pour supprimer une ligne de programme, il faut :
   - ☐ Frapper son numéro, puis faire RETURN
   - ☐ Frapper son numéro, puis faire CTRL/X

6. L'ordre NEW sert à :
   - ☐ Relancer l'exécution d'un programme
   - ☐ Effacer la mémoire

7. Pour effacer l'écran, il faut faire :
   - ☐ CLS
   - ☐ CTRL/L
   - ☐ ? CHR$(12)

8. Si le programme doit demander à l'opérateur de nouvelles données, vous devrez introduire une instruction :
   - ☐ ENTER
   - ☐ INPUT

9. Un organigramme est :
   - ☐ La représentation graphique d'un algorithme
   - ☐ Le dessin d'un programme

10. Pour interrompre un programme en cours d'exécution, vous ferez :
   - ☐ CTRL/C
   - ☐ STOP

11. Pour créer une temporisation de cinq secondes, vous écrirez :
   - ☐ STOP 5
   - ☐ WAIT 500

12. L'ordre EDIT 20
   - ☐ Supprime la ligne 20
   - ☐ L'affiche sur l'écran pour corrections

Réponses page 72

1

UNE ADDITION QUI N'EN FINIT PAS

CLS INPUT PRINT GOTO WAIT

La scène se passe à la sortie d'un magasin : un client qui vient de faire ses achats s'aperçoit qu'il a oublié son carnet de chèques et que son porte-monnaie risque de ne pas suffire à tout payer.

La caissière tape alors les prix en faisant afficher au fur et à mesure la somme due, jusqu'à ce que le client lui crie « halte ».

```
10 CLS
20 S=0
30 INPUT P
40 S=S+P
50 PRINT S
60 GOTO 30
```

REMARQUES

10 — on efface l'écran

20 — on efface l'ardoise du client

30 à 60 — c'est le serpent qui se mord la queue : une boucle

Le spectacle suivant se déroule devant les yeux du client inquiet :

```
? 23
23
? 34
57
? 12
69
? 42
111
? 56
167
? 192
359
? 54
413
?

Ready
```

Pour arrêter on a tapé CTRL C

Il faut bien avouer qu'on peut faire mieux comme affichage, en étant plus bavard : un petit message pour la caissière, un autre pour le client, à condition de modifier deux lignes.

```
10 CLS
20 S=0
30 INPUT"PRIX D'UN ARTICLE";P
40 S=S+P
50 PRINT"SOMME DUE ";S
60 GOTO 30
```

```
PRIX D'UN ARTICLE? 23
SOMME DUE 23
PRIX D'UN ARTICLE? 34
SOMME DUE 57
PRIX D'UN ARTICLE? 56
SOMME DUE 113
PRIX D'UN ARTICLE? 12
SOMME DUE 125
PRIX D'UN ARTICLE?

Ready
```

Comment savoir à quel article on en est ? Il suffit de les compter et d'afficher ce numéro.

Mais l'instruction INPUT ne sait écrire que des messages figés. Il faut donc distinguer l'affichage du message destiné à la caissière précisant le numéro de l'article et l'introduction du prix :

```
10 CLS
15 N=1
20 S=0
25 PRINT"ARTICLE NO ";N;
30 INPUT P
40 S=S+P
50 PRINT"SOMME DUE ";S
55 N=N+1
60 GOTO 25
```

REMARQUES

25 — affichage du message

30 — introduction du prix

```
ARTICLE NO 1 ? 8
SOMME DUE 8
ARTICLE NO 2 ? 4.5
SOMME DUE 12.5
ARTICLE NO 3 ? 67.35
SOMME DUE 79.85
ARTICLE NO 4 ? 122
```

```
SOMME DUE 201.85
ARTICLE NO 5 ?

Ready
```

L'écran se remplit vite de phrases et de nombres et si on veut y voir clair il vaut mieux ne garder que ce qui concerne l'article en cours. On nettoiera l'écran entre deux articles. Mais attention il faut avoir le temps de lire la somme avant qu'elle ne s'efface ! L'instruction WAIT réalisera ce délai ajustable au réflexe humain :

```
50 PRINT"SOMME DUE ";S
55 N=N+1
56 WAIT200
57 CLS
60 GOTO 25
```

REMARQUES

56 — attente : 2 secondes

57 — effaçage

On a maintenant un écran qui contient seulement deux lignes comme celles-ci avant de s'effacer :

```
ARTICLE NO 3 ? 45.50
SOMME DUE 89.5
```

La numérotation des lignes de ce programme est irrégulière puisqu'il a été écrit par retouches successives, ce qui est un cas très réaliste.

2

UN CHRONOMÈTRE DIGITAL

CLS PRINT WAIT GOTO

Voici une première forme de programme de chronométrage à la seconde. On ne peut guère faire plus simple : c'est une application de l'instruction WAIT.

WAIT 100 provoque une pause d'une seconde, et pourtant nous avons dû mettre WAIT 95. Pourquoi ? La portion du programme qui est répétée chaque seconde ne contient pas que cette instruction. L'exécution des lignes 20, 30, 40 et 50 prend au total 5/100 seconde environ.

```
10 S=0
20 CLS
30 PRINT S
40 S=S+1
50 WAIT 95
60 GOTO 20
```

REMARQUES

10 — point de départ

20 à 60 — boucle répétée chaque seconde

50 — délai : 95/100 seconde

Nous ne reproduisons pas d'exemple d'utilisation : les secondes vont défiler en haut de l'écran, et vous pourrez vérifier vous-même que le chronométrage est précis à moins d'une seconde par minute. Nous ferons mieux (plus complet et plus précis) au chapitre VII.

3

CARACTÈRES ET CODES : LA NORME ASCII

CLS CHR$ ASC PRINT GOTO WAIT

PRINT CHR$(12) signifie afficher le caractère dont le code est 12 et nous venons de voir que ceci avait pour effet d'effacer l'écran et de remettre le curseur en haut à gauche. Ce code 12 correspond à une norme très répandue qui est valable aussi pour les imprimantes et signifie passer à la page suivante. Il y a ainsi 256 codes auxquels correspondent 256 caractères ou effets spéciaux. Entre 32 et 127 on retrouve à peu près partout les mêmes caractères selon la norme de codification ASCII. Ainsi le code de la lettre A est 65, pour B c'est 66...

Vous trouverez en fin d'ouvrage la table pour les codes 32 à 127. Mais vous pouvez interroger l'ordinateur pour connaître cette table :

PRINT CHR$(65) vous donnera A,

et à l'inverse PRINT ASC("A") affichera 65.

Affichage d'une partie de la table

On peut écrire par exemple, le programme de consultation à partir du code 32 (en-dessous les effets spéciaux sont très perturbants) :

```
10 C=32
20 PRINT C,CHR$(C)
30 C=C+1
40 GOTO 20
```

REMARQUES

10 — code du premier caractère

20 à 40 — boucle

20 — affichage du code puis du caractère

30 — code suivant.

```
32
33    !
34    "
35    #
36    $
37    %
 .    .
 .    .
 .    .
 .    .
 .    .
 .    .
 .    .
```

Au cours du spectacle, ne soyez pas inquiets pour votre ordinateur ou votre téléviseur : tout s'arrange à la fin ! Eh oui, cette boucle qui ne devrait jamais s'arrêter se bloque grâce à une erreur : ?Illegal quantity error in 20 : on n'a pas le droit de demander le caractère dont le code est 256 (ou plus). Ouf !

```
250    z
251    {
252    |
253    }
254    ~
255
256
?ILLEGAL QUANTITY ERROR IN 20
Ready
```

Si vous trouvez que la table défile trop vite, ajoutez une temporisation à la ligne 35, et pour mieux isoler les différents effets effacez l'écran :

```
30 C=C+1
35 WAIT 30
36 CLS
40 GOTO 20
```

Entre les codes 32 et 126 on obtient la liste des caractères normaux disponibles sur l'ORIC, en commençant par l'espace.

Entre 127 et 159, les résultats sont parfois surprenants : certains codes ont des effets sur la couleur du fond ou de l'encre, d'autres sur la synchronisation de l'image ou encore masquent une partie de l'écran. On obtient même des hiéroglyphes étranges venus d'ailleurs. Nous aurons l'occasion de revenir sur ces codes spéciaux.

Si on ne connaît pas le code du premier caractère souhaité

Comment démarrer la consultation à la lettre A, par exemple, si l'on ne connaît pas le code de cette lettre ? En le faisant calculer par le programme :

```
10 C=ASC("A")
20 PRINT  CHR$(C),C
30 C=C+1
35 WAIT 30
36 CLS
40 GOTO 20
```

Ici on écrit d'abord le caractère puis son code

REMARQUES

10 — code du premier caractère affiché

20 à 40 — boucle

20 — affichage du caractère puis de son code

Le spectacle change légèrement dans la région 129 à 143 et on observe l'effet des caractères spéciaux sur leur propre code.

4

L'INITIALE D'UN MOT
LEFT$ LEN

LEFT$

Certaines fonctions du Basic permettent de se livrer à toutes sortes d'opérations chirurgicales sur les chaînes de caractères. Commençons par l'extraction de la partie gauche d'un mot ou d'un groupe de mots grâce à la fonction LEFT$ (de l'anglais left : gauche). N'oubliez pas le dollar car le résultat fourni par LEFT$ est une chaîne de caractères

LEFT$(« NON »,2) vaut « NO »

et si C$ vaut « PARISIEN »,

LEFT$(C$,5) vaut « PARIS »

Vous avez compris, le deuxième nombre indique combien de caractères on souhaite extraire, et on commence par le plus à gauche.

Utilisons cette fonction pour connaître l'initiale d'un mot.

```
10 INPUT"TAPER UN MOT";M$
20 PRINT"L'INITIALE EST ";LEFT$(M$,1)
```

Un exemple d'utilisation :

```
RUN
TAPER UN MOT? ZINC
L'INITIALE EST Z

Ready
```

Pour l'instant l'ordinateur ne nous apporte rien. Ça deviendrait plus intéressant si l'on avait emmagasiné en mémoire un tas de noms et qu'on veuille les ressortir par leur initiale. Patience !

RIGHT$

L'équivalent de LEFT$ pour la partie droite s'appelle RIGHT$ (right signifie droite en anglais), par exemple :

RIGHT$ (« BONJOUR »,4) vaut « JOUR » c'est-à-dire les 4 derniers caractères.

LEN

Si on a besoin de connaître le nombre de caractères d'un mot ou d'un groupe de mots, c'est la fonction LEN (du mot anglais Length : longueur) qui va vous renseigner : LEN (« OUI ») vaut 3.

```
W$="ANTICONSTITUTIONNELLEMENT"

PRINT LEN(W$)
25
```

5

UN TÊTE A QUEUE VERBAL
MID$

MID$

Comment écrire à l'envers : de droite à gauche ? Il suffit de parcourir le mot ou la phrase caractère par caractère, en commençant par le dernier. L'affichage se terminera faute de caractère, parce que nous n'avons pas encore vu comment finir plus joliment ! La fonction à tout faire sur une chaîne de caractères s'appelle MID$ (de l'anglais middle : milieu) et elle permet d'analyser un fragment de chaîne situé n'importe où.

MID$("TARABISCOTTE",5,3) vaut "BIS"

parce que B est le *cinquième* caractère de la chaîne de départ et qu'on a pris *trois* caractères.

MID$("TARABISCOTTE",5) donne "BISCOTTE" : le nombre de caractères à extraire n'est pas précisé si MID$ va jusqu'au bout. Pour isoler le caractère numéro P d'une chaîne située dans M$, il suffit d'écrire MID$(M$,P,1).

```
10 INPUT"TEXTE A INVERSER";T$
20 C=LEN(T$)
30 CLS
40 PRINT MID$(T$,C,1);
50 C=C-1
60 GOTO 40
```

REMARQUES

20 — on commence par le dernier caractère

50 — on régresse

Dans l'exemple suivant :

```
TEXTE A INVERSER? ARROSOIR


RIOSORRA
?ILLEGAL QUANTITY ERROR IN 40
Ready
```

l'exécution se termine sur le message ?ILLEGAL QUANTITY ERROR au moment où on cherche à extraire le caractère situé avant le premier (C = 0) .

6

UN COMPTE ROND
STR$ VAL

Grâce aux fonctions LEFT$, RIGHT$, MID$ et LEN, la manipulation des chaînes de caractères est plus commode que celle des nombres. Mais on peut l'appliquer aux nombres en utilisant la fonction STR$ qui convertit un nombre en chaînes de caractères. Pourquoi pas ?!...

STR$(1357) vaut ''1357'' et on pourra toujours faire le trajet inverse pour retrouve un « vrai » nombre :

STR$

VAL('' – 285'') donne le nombre – 285

VAL

Par exemple, si on souhaite arrondir un prix donné en francs, à la dizaine de francs supérieure, il suffit de changer le chiffre de droite pour un zéro et d'augmenter de 1 son voisin (le chiffre des dizaines).

On peut écrire une première version du programme.

```
 10 CLS
 20 INPUT"NOMBRE A ARRONDIR";N
 30 PRINT SPC(22);"CHIFFRES"
 40 PRINT SPC(13);"DE DROITE DIZAINE UNIT
E"
 50 N$=STR$(N)
 60 L=LEN(N$)
 70 U$=RIGHT$(N$,1)
 80 D$=MID$(N$,L-1,1)
 90 N$=LEFT$(N$,L-2)
100 PRINT"AU DEPART :"
110 PRINT SPC(16);N$,,D$,,U$
120 R=INT(VAL(U$)/5)
130 D$=CHR$(ASC(D~)+R)
140 PRINT"RETENUE SUR LES DIZAINES ";R
150 PRINT SPC(16);N$,,D$,,U$
160 U$="0"
170 PRINT"REMISE A ZERO DE L'UNITE"
180 PRINT SPC(16);N$,,D$,,U$
190 N$=N$+D$+U$
200 PRINT"CE QUI DONNE : ";N$
```

REMARQUES

20 — introduction du nombre à arrondir

50 — conversion en chaîne de caractères

70 — le chiffre des unités

80 — celui des dizaines

90 — les autres

120 — la retenue qu'il faut ajouter au chiffre des dizaines (0 ou 1)

190 — reconstruction du nombre complet (toujours sous forme de chaîne de caractères)

Exemple :

```
NOMBRE A ARRONDIR? 1234
                        CHIFFRES
              DE DROITE DIZAINE UNITE
AU DEPART :
                 12       3       4
RETENUE SUR LES DIZAINES 1
                 12       4       4
REMISE A ZERO DE L'UNITE
                 12       4       0
CE QUI DONNE : 1240

Ready
```

Si on désire arrondir à la dizaine la plus proche, il faut modifier la ligne 120 :

```
10 CLS
20 INPUT"NOMBRE A ARRONDIR";N
30 PRINT SPC(22);"CHIFFRES"
40 PRINT SPC(13);"DE DROITE DIZAINE UNIT
E"
50 N$=STR$(N)
60 L=LEN(N$)
70 U$=RIGHT$(N$,1)
80 D$=MID$(N$,L-1,1)
90 N$=LEFT$(N$,L-2)
100 PRINT"AU DEPART :"
110 PRINT SPC(16);N$,,D$,,U$
120 R=INT(VAL(U$)/5)
130 D$=CHR$(ASC(D$)+R)
```

si le chiffre des unités est inférieur à 5 la retenue R vaudra 0 sinon ce sera 1.

```
NOMBRE A ARRONDIR? 1234
                        CHIFFRES
              DE DROITE DIZAINE UNITE
AU DEPART :
                 12       3       4
RETENUE SUR LES DIZAINES 0
                 12       3       4
REMISE A ZERO DE L'UNITE
                 12       3       0
CE QUI DONNE : 1230

Ready
```

```
NOMBRE A ARRONDIR? 1236
                          CHIFFRES
              DE DROITE DIZAINE UNITE
AU DEPART :
                  12       3       6
RETENUE SUR LES DIZAINES 1
                  12       4       6
REMISE A ZERO DE L'UNITE
                  12       4       0
CE QUI DONNE : 1240

Ready
```

On retrouvera ces manipulations de chaînes de caractères ainsi que les instructions INPUT WRITE GOTO dans les chapitres suivants, Mais l'accent sera mis sur les instructions nouvelles.

REPONSES AUX QUESTIONS SUR LE CHAPITRE IV

1. Non, il n'y a pas d'erreur, ici. Vous pouvez attribuer aux lignes n'importe quel numéro : respectez seulement un ordre croissant.

2. L'exécution d'un programme est commandée par RUN. L'ordre GO n'existe pas, tel, de toute façon (sauf sous des formes telles que GOTO).

3. Le point-virgule, dans un ordre PRINT, supprime le retour à la ligne. L'affichage se poursuivra sur la même ligne.

4. Oui, l'ordre LIST ré-affiche le programme en mémoire.

5. Pour supprimer une ligne de programme, il faut frapper son numéro puis RETURN. Rappelez-vous que CTRL/X sert à abandonner ce qu'on vient de frapper ; la machine n'en tiendra pas compte ; par conséquent, il n'y aurait pas suppression.

4. NEW sert à effacer la mémoire (vive, centrale), qui contient le programme, les données et les résultats.

7. Ces trois ordres effacent l'écran : à vous de choisir celui que vous préférez ! Rappelez-vous que la frappe de CTRL/L ne peut se faire qu'en mode direct. Par contre, CLS et ? CHR$(12) peuvent intervenir en mode direct ou en mode programmé. On préfèrera CLS, tout simplement parce que c'est plus court à frapper et plus évident (si vous vous souvenez de sa signification anglaise : « *Clear Screen* », soit « effacer l'écran »).

8. C'est INPUT qui sert à introduire de nouvelles données en cours de l'exécution d'un programme (on a inventé, ici, le mot ENTER pour les besoins de la question).

9. Ici, on a répété deux fois la même chose, d'abord de façon « savante », puis très simplement. En effet, un « algorithme » est la « logique » du programme, la façon dont il est organisé pour atteindre le résultat voulu. Et « représentation graphique » ou « dessin » sont très synonymes.

10. Pour interrompre un programme, il faut faire CTRL/C.

11. Vous créerez une temporisation de cinq secondes avec WAIT 500.

12. L'ordre EDIT 20 affiche la ligne 20 sur l'écran afin que vous puissiez la corriger.

CHAPITRE V

COULEURS ET SONS PROGRAMMÉS

On va enfin vous présenter la gamme des couleurs de l'Oric – 1. Avec, de plus, les sons programmés, au nombre de quatre : une clochette, un tir d'arme galactique à laser (eh oui !), un coup de feu et une explosion. Pour les créer, il nous faut vous initier aux boucles du type FOR... TO et aux nombres aléatoires. Mais votre peine va aussitôt trouver sa récompense, croyez-le.

Principaux thèmes étudiés

Boucles	Emboîtage des boucles
FOR... TO	Règles avec les boucles
NEXT	Sons
Organigramme de boucle FOR... TO	CTRL/G
Instructions multiples	PING
Double-point	ZAP
Pas dans FOR... TO	SHOOT
STEP	EXPLODE
Décomptage avec FOR... TO	Nombres aléatoires
Couleurs	RND
PAPER	Notation scientifique
INK	Entiers
Remarques	INT
REM	Bataille galactique
Défilé des couleurs	RND(Ø)
	RND(– N)

1. Boucles avec les ordres FOR... NEXT

BOUCLES

On a vu, dans le chapitre précédent, que l'ordre GOTO pouvait commander un « bouclage ». L'ordre FOR... NEXT permet également la création de boucles, mais avec deux différences essentielles :

1) — Voici le principal : vous contrôlez le *nombre de boucles* à effectuer, puisque FOR... TO en a fixé les limites.

FOR... TO

NEXT

2) — Puis, mais c'est le moins important pour notre action, cette instruction se compose en réalité de deux ordres distincts. L'un est FOR... TO ce qui signifie « pour... jusqu'à » et sert à fixer des limites, le second est NEXT (« au suivant ! »), qui renvoie à la valeur suivante. Il faudra donc *deux* lignes.
C'est facile à vérifier avec ce court programme qui va afficher les trois chiffres 1, 2 et 3. Entrez-le en machine et exécutez-le pour voir ce que cela donne ; on vous l'expliquera aussitôt après ; on a fait suivre l'exécution, ici :

```
10 FOR N=1 TO 3
20 ? N
30 NEXT N
40 ? "C'est fini!"

RUN
1
2
3
C'est fini!

Ready
```

Que s'est-il passé ? A la ligne 10, on a fixé les limites de N, qui pourra aller de 1 à 3, en commençant par 1.

Le comptage se fait par unités successives : 1, puis 2, puis 3... A moins qu'on n'ajoute des ordres contraires, comme on va le voir plus loin.

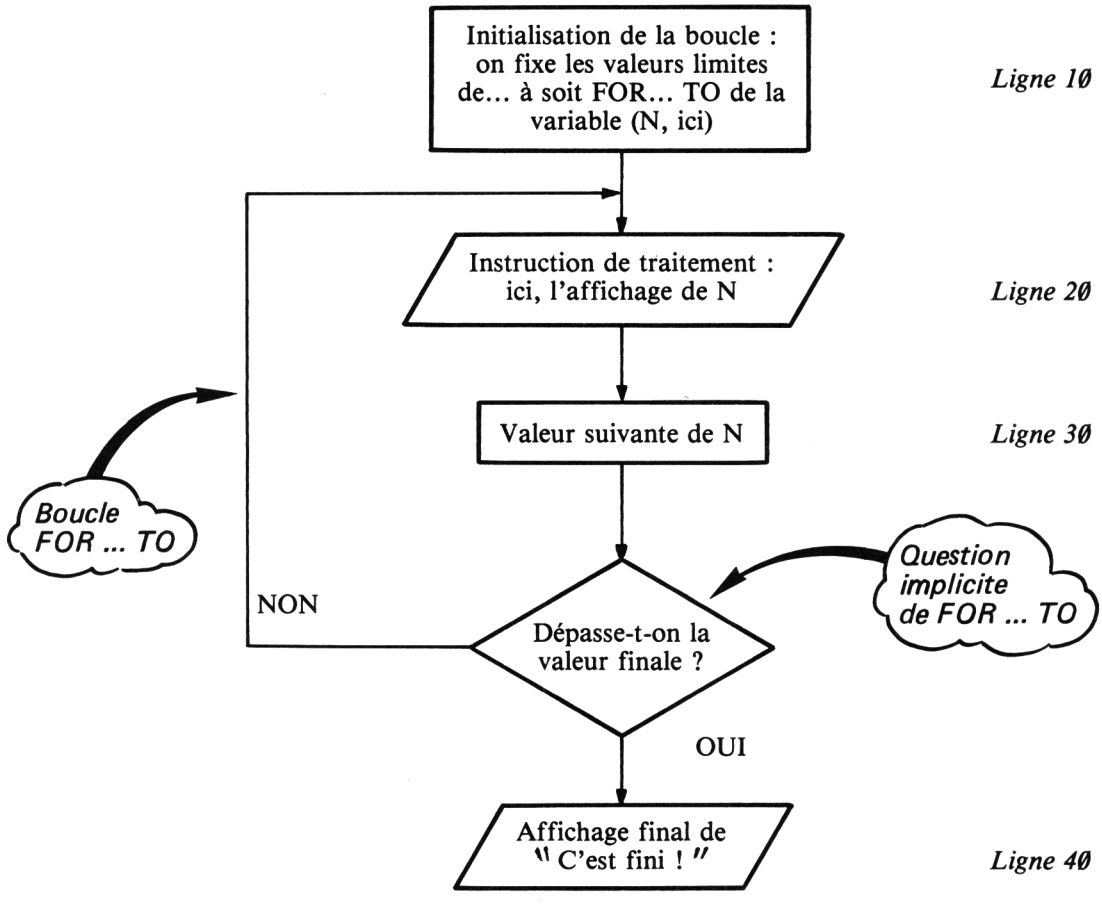
Puis, en 20, on commande l'affichage de N, donc de 1 au premier tour. Le programme rencontre alors la ligne 30 qui lui demande le N suivant ; il se branche alors à nouveau en 10 et, s'apercevant que la valeur 1 a déjà été utilisée, prend la 2 et l'attribue à N. En 20, il affiche 2, Puis passe en 30 qui renvoie en 10 où N prend la valeur 3. Celle-ci est affichée à son tour, puis 30 renvoie encore en 10 mais là, le programme s'aperçoit que toutes les valeurs possibles ont été utilisées. Par conséquent, il saute toute la chaîne et poursuit à l'instruction suivante qui est la 40 ; il affiche alors « C'est fini ! ».

Lorsque toutes les valeurs permises ont été utilisées, le programme passe à l'instruction qui suit la ligne NEXT.

ORGANIGRAMME DE LA BOUCLE FOR... TO

On peut illustrer ce processus à l'aide de l'organigramme ci-contre.
Remarquez le losange de l'organigramme : il pose une question, contenue implicitement dans l'ordre FOR... TO : *la valeur finale est-elle dépassée ?* si *non,* on boucle ; si *oui,* on passe à l'instruction suivante. On emploie donc deux sommets de ce losange, selon que la réponse est NON ou OUI. Chaque fois qu'on posera une question, on l'insérera dans un losange.
L'instruction de traitement contenue dans la boucle peut fort bien faire appel à autre chose que la variable contenue dans la boucle FOR... TO ; modifiez simplement le programme précédent à la ligne 20 pour obtenir :

```
10 FOR N=1 TO 3
20 ?"OUI,";
30 NEXT N
40 ? "C'est fini!"

RUN
OUI,OUI,OUI, C'est fini!
```



Organigramme de la boucle FOR... TO

Comment faire pour écrire C'est fini ! sur la ligne suivante ? Réfléchissez un instant : il faut *annuler* l'effet du point virgule prédécent, de la ligne 20, mais *seulement* lorsqu'on sort de la boucle. Qu'à cela ne tienne : on va placer un PRINT rien du tout, *sans* point-virgule final, lui ; on obtiendra cet affichage sur *la même ligne* que les OUI,OUI,OUI, mais il provoquera un retour à la ligne pour l'affichage final. Cela donne, en récapitulant :

Nous avons mis PRINT en toutes lettres en supposant que vous avez fait un LIST

```
10 FOR N=1 TO 3
20 PRINT "OUI,";
30 NEXT N
35 PRINT
40 PRINT "C'est fini!"

RUN
OUI,OUI,OUI,
C'est fini!
```

Ajouter cette instruction ...

... Et l'affichage final se fera sur la ligne suivante

Et comment pourriez-vous faire pour séparer d'une ligne vide les « OUI,OUI,OUI », de « C'est fini ! » ? On nous l'a déjà dit dans le chapitre précédent, rappelez-vous : placez un autre PRINT :

```
10 FOR N=1 TO 3
20 PRINT "OUI,";
30 NEXT N
35 PRINT
36 PRINT
40 PRINT "C'est fini!"

RUN
OUI,OUI,OUI,

C'est fini!
```

Ce PRINT commande ...

... Ce saut de ligne

Mais pourquoi employer *deux* lignes, 35 et 36, qui ne comprennent pas grand chose chacune ? Ne pourrait-on les regrouper ? Oui, le Basic accepte qu'on place *plusieurs* instructions sur une unique ligne (avec un *unique numéro*).

INSTRUCTIONS MULTIPLES

DOUBLE-POINT

Pour séparer des instructions sur une ligne, on place un double-point (:). C'est le « séparateur » officiel.

Aussi, on peut ré-écrire encore ainsi le programme précédent, en cumulant même 3 lignes. Commencez par *supprimer* 35 et 36, puis refrappez ainsi la ligne 40.

3 instructions sur une seule ligne

Ne vous égarez pas dans tous ces : ?

Les : servent de séparateurs

```
40 ?:?:? "C'est fini!"
```

Un LIST vous donnera le programme complet suivant :

```
10 FOR N=1 TO 3
20 PRINT "OUI,";
30 NEXT N
40 PRINT:PRINT:PRINT  "C'est fini!"
```

Le RUN, bien entendu, procurerait le même résultat que ci-dessus.

PAS DANS FOR... TO

STEP

2. Boucles FOR... TO avec un pas

On a vu que la boucle FOR... TO « incrémentait » la variable d'un *pas positif unitaire :* la variable N passait successivement à 1, puis 2, puis 3. Or, on peut lui imposer un *pas* différent, par exemple un demi, ce qui s'écrit 0,5 en français courant.

Or, rappelez-vous que le Basic, d'origine anglaise, place un « point » décimal là où nous mettons une « virgule » décimale. Donc, au lieu d'écrire Ø,5 on écrira Ø.5.
Bien mieux, le Basic estime que le Ø avant le point est encore superflu. On peut donc se contenter d'écrire .5.

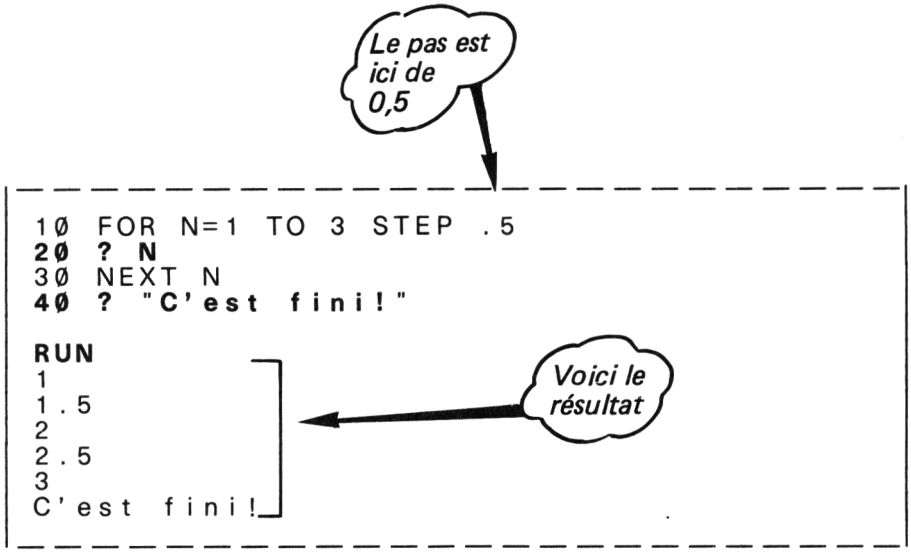
Si vous placez cependant un Ø, cela ne constitue pas une faute.
Avez-vous conservé en mémoire le programme précédent ? Si oui, commencez par modifier la ligne 1Ø pour obtenir :

```
1Ø FOR N=1 TO 3 STEP .5
```

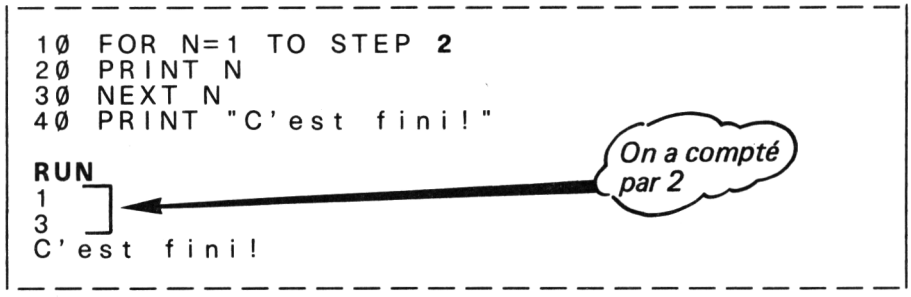
Pour apporter cette correction, on va employer EDIT. Faites successivement.

- EDIT 1Ø, puis RETURN. La ligne 1Ø apparaît sur l'écran.
- Avec CTRL/A, amener le curseur après le 3 de "FOR N=1 TO 3".
- Frappez alors STEP .5, puis RETURN.

La correction est achevée. Modifiez alors les lignes 2Ø et 4Ø pour obtenir :



Vous pourriez choisir un pas absolument quelconque, par exemple de 2. Modifiez à nouveau la ligne 1Ø ainsi, et lancez le programme ; cela donnera :



DECOMPTAGE AVEC FOR... TO

Le pas peut être négatif, si l'on veut *décompter,* mais il faudra alors aller de *3 à 1* et non plus de 1 à 3 ; il suffit, à nouveau, de ne modifier que la seule ligne 1Ø :

Veillez à bien écrire : de 3 à 1

```
1Ø FOR N=3 TO 1 STEP -1
2Ø PRINT N
3Ø NEXT N
4Ø PRINT "C'est fini!"

RUN
3
2
1
C'est fini!
```

Le pas est négatif

Vous pouvez combiner tout ceci à volonté, pas quelconques, positifs ou négatifs. Faites quelques expériences par vous même, pour bien « vivre » cet ordre.

3. Avec 8 couleurs

COULEURS

Votre Oric – 1 dispose de 8 couleurs, le blanc et le noir compris. Chaque couleur se voit affecter un numéro et c'est par ce numéro qu'on l'appellera. Il s'agit de :

COULEUR	NUMERO
Noir	0
Rouge	1
Vert	2
Jaune	3
Bleu	4
Magenta	5
Cyan	6
Blanc	7

C'est un bleu-vert

PAPER

INK

Vous pouvez mettre en couleur la « page de texte », l'équivalent de la feuille qui apparaît sur l'écran et s'appelle en langage Oric « PAPER », ou le texte lui-même : INK (pour *encre*).
Commencez par mettre la page en rouge. Faites :

```
PAPER 1
```

Avec RETURN, la «page» va passer au rouge

Mettez maintenant le texte affiché en vert avec :

```
INK 2
```

Avec RETURN, le texte se met en vert

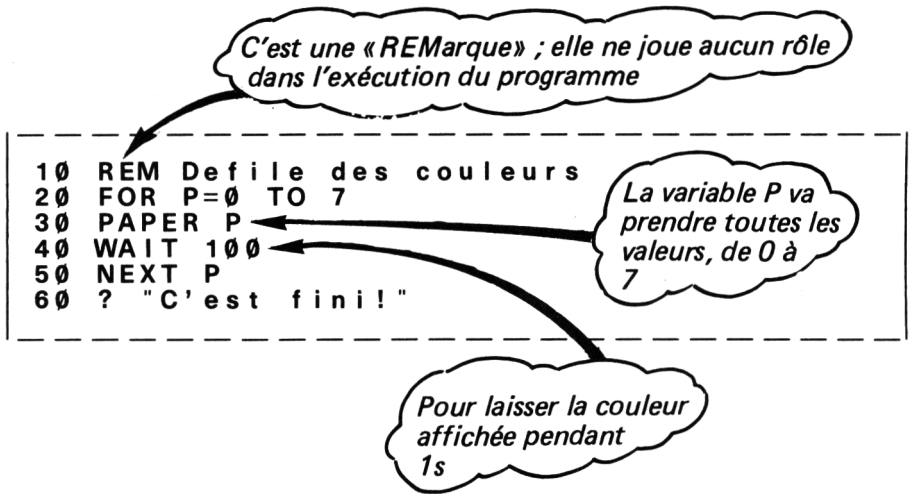
Ne soyez pas trop exigeant avec votre téléviseur couleur : les couleurs pourront « baver » un peu, ou ne pas être rigoureusement celles annoncées. Après tout, qu'importe si le résultat reste intéressant ? Nul n'est parfait, encore moins les réglages de votre téléviseur.

REMARQUES
REM

Voici, maintenant, un petit programme destiné à faire défiler les 8 couleurs, chacune restant affichée une seconde, en « page ». Rappelez-vous que lorsque l'*encre* et la *page* sont de la même couleur, l'affichage se perd, devient invisible. Ce programme emploie une boucle FOR... TO. En outre, on lui a ajouté un titre, sous forme de *remarque.*

L'ordre REM, pour « remarque », sert à placer des explications pour celui qui lit le programme, et uniquement pour lui. Lorsque le Basic rencontre l'ordre REM, il saute tout ce qui suit et passe à la ligne suivante.

Voici ce programme :



Faites un RUN : les 8 couleurs vont se succéder. Si vous voulez que le processus continue, remplacez 60 par :

```
60 GOTO 20
```

La boucle sera bouclée à perpétuité, ou jusqu'à un CTRL/C.

Compliquons le jeu en faisant varier la couleur de la feuille (PAPER) mais aussi celle du texte (INK), à l'aide de deux boucles, cette fois : une boucle P pour PAPER, et une boucle I pour INK. Le principe est le suivant : on affiche une page d'une couleur donnée, puis on fait défiler les couleurs du texte ; lorsqu'elles sont toutes passées, on change la couleur de la page et on recommence ; etc. *Etudiez bien ce programme pour comprendre le fonctionnement de ces deux boucles,* car c'est très important ; c'est le seul concept nouveau que nous voulons d'ailleurs vous présenter ici.

Plus exactement, le processus sera le suivant ;

- La page passe en noir (Ø)
  - — le texte passe en noir (Ø)
  - — En rouge (1)
  - — En vert (2)
  - — En jaune (3)
  - — En bleu (4)
  - — En magenta (5)
  - — En cyan (6)
  - — En blanc (7)
- La page passe en rouge (1)
  - — Le texte passe en noir Ø
  - — En rouge (1)

On a là une sorte de «petite» boucle interne ...

Et par opposition, une «grande» boucle externe

Ici, répétition de la «petite» boucle interne

... Etc. !

«Grande» boucle : PAPER

Le programme est ainsi :

```
10 REM Defile des couleurs
20 FOR P=Ø TO 7
30 FOR I=Ø TO 7
40 PAPER P
50 INK I
60 WAIT 1ØØ
70 NEXT I
80 NEXT P
90 INK Ø
1ØØ ? "C'est fini!"
```

Tiens, pourquoi ?

«Petite» boucle : INK

Lancez ce programme : vous verrez se succéder les pages de couleurs, puis, et pour chaque page, les textes en couleurs. A la fin, en ligne 9Ø, on a placé un INK Ø, pouvez-vous deviner pourquoi ? La dernière « image » couleur de ce programme donne INK 7 et PAPER 7, donc du texte blanc sur fond blanc ! Vous ne lirez pas grand chose. Aussi a-t-on rétabli l'*encre* en noir (Ø).

Notez ce point capital : les deux boucles sont « emboîtées », la « petite » est intégralement contenue dans la « grande », ce que l'on a marqué sur le listage du programme.

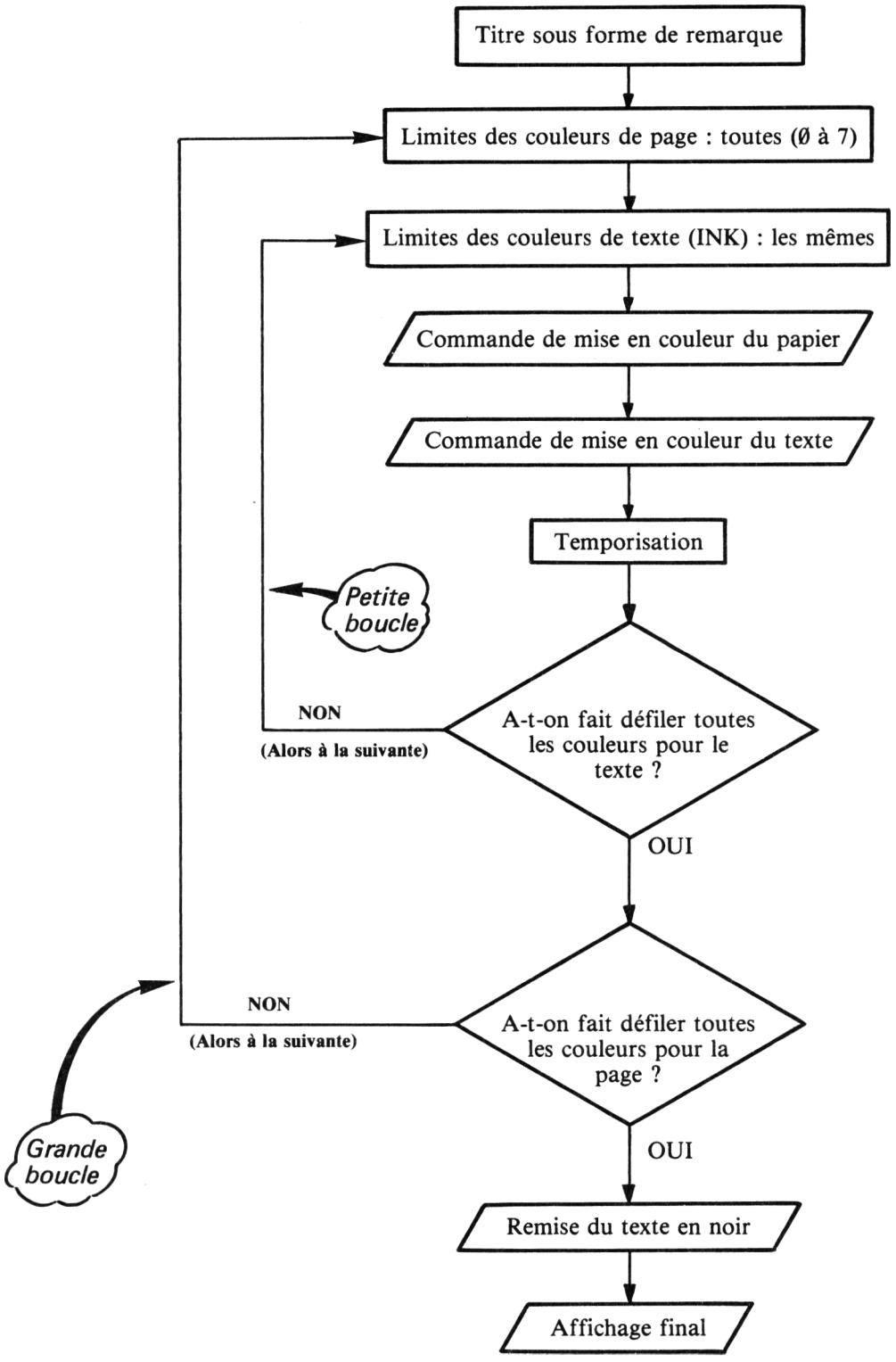
L'organigramme du programme (ci-contre) montre bien, lui aussi, la présence de ces deux boucles.

EMBOITAGE DES BOUCLES

A ne jamais oublier

Deux boucles (deux, ou davantage) doivent être soit totalement emboîtées (contenues l'une dans l'autre), soit totalement distinctes.

On peut illustrer cela ainsi :



Organigramme de la double boucle de couleurs

Autorisé	*Interdit !*
Boucles emboîtées	Pas de chevauchement, SVP !
Boucles distinctes	

REGLES AVEC LES BOUCLES

Cela signifie encore que, si on ouvre la boucle A, puis on démarre la boucle B, *il faut* refermer d'abord B (avec un NEXT B) puis *après* seulement A (avec un NEXT A). Sinon, vous commetteriez cette erreur typique des débutants,
Revoyez le programme ci-dessus : on a *ouvert* (FOR... TO) la boucle P, puis I, mais on a refermé d'*abord* I (avec le NEXT de la ligne 7Ø) puis P (avec le NEXT de 8Ø). Il n'aurait pas fallu faire l'inverse : attention à ce type d'erreur !

Nous supposons que vous multipliez maintenant les essais. Si vous sentez que vous perdez la maîtrise de l'Oric, qu'il « ne veut plus vous obéir », c'est que vous avez commis une erreur quelque part, ou une fausse manipulation. Dans ce cas, n'hésitez pas à faire un « RESET ».

Pour cela, retournez l'Oric – 1 et insérez la pointe d'un crayon dans la grande fenêtre qui apparaît en-dessous. Pressez le poussoir (que vous ne distinguerez pas : dommage !). Vous provoquez ainsi ce redémarrage qu'on appelle le RESET (« remise en condition de départ »). Il laisse intact le contenu des mémoires centrales. On l'appelle encore un *redémarrage à chaud.*
N'oubliez pas, non plus, de faire NEW pour effacer la mémoire lorsque vous redoutez qu'un nouveau programme n'interfère fâcheusement avec un ancien.

4. Des sons aussi

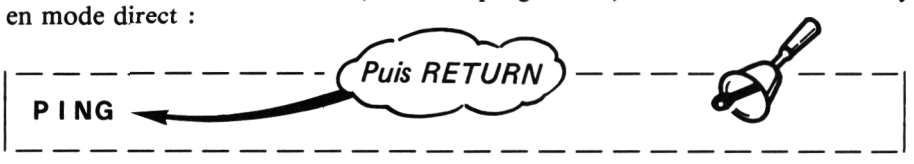
SONS

Votre Oric dispose de la couleur (on vient de le voir), sait dessiner (patience : d'autres chapitres suivent !), et produit des sons ou même de la musique. On va examiner ici les sons, laissant la musique à un prochain chapitre.

CTRL/G

PING

Vous avez d'abord remarqué que CTRL/G produisait un tintement de clochette. Vous obtiendrez le même effet, dans un programme, en écrivant PING. Essayez, en mode direct :



Chaque touche est, en outre, sonorisée. Vous pouvez supprimer ce son en faisant CTRL/F, ou le rétablir avec la même combinaison (vérifiez-le).

Mais voici mieux :

ZAP
SHOOT
EXPLODE

• L'ordre ZAP produit un sifflement décroissant sensé imiter une arme galactique à laser. C'est une simple question d'imagination !

• L'ordre SHOOT simule le bruit d'un coup de fusil.

• Enfin, EXPLODE, comme son nom l'indique, simule une explosion.

Expérimentez ces ordres en mode direct, puis organisez une fusillade galactique avec la séquence suivante :

Pour 10 tirs

```
10 FOR N=1 TO 10
20 ZAP
30 WAIT 8
40 NEXT N
```

C'est le son de l'arme galactique

Avec le RUN, vous entendrez dix tirs galactiques, très réguliers. Trop, même : il faudrait qu'ils surviennent un peu à l'improviste, au hasard ; mais ça aussi, votre ordinateur sait le faire.

5. Nombres au hasard et bataille galactique

NOMBRES ALEATOIRES
RND

Une fonction spéciale du Basic sert à tirer des nombres au hasard, c'est RND ; ces lettres proviennent du mot anglais « *random* », soit *aléatoire,* au hasard. L'ordre RND demande à être suivi par un nombre *absolument quelconque* mais placé entre parenthèses ; le Basic a, parfois, de telles fantaisies... Ne mettez ni zéro, ni nombre négatif, et par paresse, utilisez le 9 qui se trouve situé sur la même touche que l'ouverture des parenthèses. Essayez simplement en mode direct :

Ordre de tirage et d'affichage d'un nombre aléatoire

```
? RND(9)
.973136996
```

L'auteur a obtenu cette valeur bizarre, un peu inférieure à 1. Vous tirerez peut-être une autre valeur

Créons un court programme pour afficher plusieurs valeurs aléatoires. Voici ce programme, suivi d'un RUN :

Pour dix valeurs aléatoires

```
10 FOR N=1 TO 10
20 ? RND(9)
30 NEXT N

RUN
.103117626
.0177148333
.779343355
.551834438
.617419111
.960296981
.547150891
.802192734
.814107273
.131137465
```

Voici ce que l'auteur a obtenu

Il est possible que vous obteniez les mêmes valeurs ; en tous cas et à chaque remise en service de l'Oric – 1, vous retrouverez la même série.

En effet, ces valeurs sont appelées « aléatoires » mais elles sont en réalité « pseudo-aléatoires ». Elles appartiennent à une très grande série répétitive de nombres, toujours inférieurs à 1.

NOTATION SCIENTIFIQUE

Parfois même, vous obtiendrez des nombres tellement petits qu'ils seront affichés en *notation scientifique.* Ce type de notation utilise les puissances ; si vous les avez étudiées, cela ne vous déconcertera pas. Ainsi, rappelez-vous qu'un 1 suivi de douze zéros (soit 1000000000000) peut s'écrire aussi 10^{12} en écriture courante. De même, un millionnième (0,000001) peut s'écrire 10^{-6}. En informatique, on écrira :

1E + 12 pour 10^{12}

1E – 06 pour 10^{-6}

La lettre E représente la « base » 10 ; le cœfficient qui suit, avec son signe, est l'*exposant.* Avant le E peuvent se trouver d'autres chiffres, avec le signe *plus* ou *moins.*

C'est ce mode d'écriture qu'on appelle donc la *notation scientifique ;* ne vous laissez pas démonter par elle et, à la rigueur, ignorez les valeurs aléatoires en notation scientifique !

Mais revenons à nos tirages : les nombres sont bien petits, et il va falloir les multiplier sérieusement pour obtenir des valeurs supérieures.

D'abord, notez que les tirages sont compris entre 0 (qui peut être tiré) et 1 (mais le 1 ne sera jamais tiré : on ira jusqu'à 0.999999999...).

Si l'on veut obtenir des valeurs comprises entre 0 et 15, par exemple, il faudra multiplier les nombres tirés par 15. Modifiez la ligne 20 du programme précédent pour introduire la multiplication par 15, et relancez-le. Vous souvenez-vous comment l'on procède pour modifier une ligne ? Utilisez LIST, ou LIST 20, ou EDIT 20 au choix. Cela donnera :

On a multiplié par 15 ici

```
10 FOR N=1 TO 10
20 PRINT RND(9)*15
30 NEXT N

RUN
7.5769316
6.04258929
14.7681638
1.90229746
8.08045929
12.0876921
9.32072512
13.1995662
4.19004824
14.9729614
```

Valeurs obtenues par l'auteur

Remarquez qu'en multipliant par 15 des valeurs comprises entre Ø (inclus) et 1 (exclu), on peut obtenir toutes les valeurs entre Ø (inclus) et 15 (exclu), donc 14,9999...

Ces nombres décimaux sont encore très embarassants. On va éliminer la partie décimale (à droite de la virgule, pardon : du point décimal) en réclamant au Basic la seule partie *entière* de ces nombres. Sachant qu'*entier* se dit « *integer* » en anglais, vous comprendrez pourquoi l'ordre se code INT.

ENTIERS

INT

Vérifiez le fonctionnement de INT en mode direct ; il exige que le nombre soit mis entre parenthèses à sa suite. Par exemple, faites :

```
? INT(123.456)
123

Ready
```

L'entier de 123, 456, est bien 123

En passant, notez une singularité de l'ordre INT : il n'arrondit pas et *fournit toujours la valeur entière immédiatement inférieure.* Par exemple (vérifiez-le) :

```
? INT(7.89)
7

Ready

? INT(-7.89)
-8
```

L'entier (immédiatement inférieur) de 7,89 est bien 7 ...

... Mais celui de - 7,89 est - 8

Refermons cette parenthèse pour en ouvrir d'autres, à la ligne 2Ø et comme annoncé. Modifiez cette ligne de façon que le programme soit le suivant, puis lancez-le :

```
1Ø FOR N=1 TO 1Ø
2Ø PRINT INT(RND(9)*15)
3Ø NEXT N

RUN
9
1Ø
8
7
1
9
2
1
14
13
```

Attention aux parenthèses : il faut toujours en refermer autant qu'on en a ouverts !

L'auteur a obtenu ces belles valeurs entières, cette fois

BATAILLE GALACTIQUE

Enfin, nous disposons de valeurs faciles à comprendre et à exploiter : on va les introduire dans notre guerre galactique pour espacer au hasard les tirs. Ce nouveau programme résulte de l'introduction d'un tirage aléatoire entre Ø et 15 dans l'ex-ordre 3Ø de temporisation du programme galactique précédent. Voici ce que cela donne :

```
10 FOR N=1 TO 10
20 ZAP
30 WAIT INT(RND(9)*15)
40 NEXT N

RUN
```

C'est déjà plus réaliste. N'hésitez pas à varier la temporisation en multipliant par 100 au lieu de 15, dans la ligne 20, par exemple, ou par d'autres valeurs.

Sachez cependant encore deux choses, à propos de l'ordre RND :

RND(0)
RND(– N)

• Si vous faites RND(0), le Basic vous renverra le dernier nombre aléatoire tiré, et non un nouveau.

• Si vous utilisez un nombre négatif, par exemple RND(–9), vous recevrez une valeur, *fixe,* et liée au nombre négatif, qui détermine le point de départ de la série. Ainsi pourriez-vous la modifier à chaque lancement.

Ainsi, avec RND(–9), l'auteur a toujours obtenu (en notation scientifique, remarquez) 3.3647666E–08. D'autres valeurs négatives fournissent d'autres nombres. Nous nous en serions voulu de ne pas vous l'avoir précisé !

QUESTIONS SUR LE CHAPITRE V

N'hésitez pas, dès maintenant, à imaginer par vous-même des courtes séquences de programmes. C'est, de loin, le meilleur exercice. Mais voyons, au préalable, si vous répondrez facilement à ces quelques questions fondamentales.

Y a-t-il ou n'y a-t-il pas d'erreur dans les lignes suivantes, extraites de leur contexte ?

1. FOR N=1 TO N=23
2. FOR N=37 TO 73
3. FOR N=37 TO 73 STEP 25
4. FOR N=37 TO 73 STEP 1ØØ
5. FOR N=37 TO 73 STEP .Ø1
6. FOR N=37 TO 73 STEP −25
7. FOR N=37 TO Ø STEP −.1
8. FOR N=1 TO 3:?N: NEXT N
9. FOR N=T+2 TO T+5:?T: NEXT T

Bon ! Si vous vous en êtes bien sorti, bravo. Continuons, alors :

10. L'ordre PAPER 3 commande :
    - ☐ Le passage en jaune (couleur 3, c'est vrai !) de la page
    - ☐ Le passage en jaune du texte
11. L'ordre INK=1 commande
    - ☐ Le passage en rouge (couleur 1, c'est vrai aussi !) de la page
    - ☐ Le passage en rouge du texte
12. La réponse de l'ordinateur à une instruction 99 REM 4*4 sera :
    - ☐ 16
    - ☐ Rien du tout !
13. L'ordre RND sert à
    - ☐ Fournir une valeur aléatoire
    - ☐ Rendre (« RND ») des couleurs ou des sons
14. Pour la machine, RND(9) et RND(18) sont-ils équivalents ?
    - ☐ Oui
    - ☐ Non

Et pour terminer, une question très difficile :

15. Vous voulez simuler un jet de dé, donc obtenir toutes les valeurs comprises entre Ø (inclus) et 6 (inclus). Quel ordre appliquerez-vous :
    - ☐ ? RND(6)
    - ☐ ? INT(RND(9)*6)+1

Réponses page 88

7

COMPTE À REBOURS
FOR NEXT PING

Dans les jeux chronométrés, il est souvent plus utile de savoir le temps qu'il reste à jouer. Il n'y a qu'à décompter, par exemple à partir de 60 jusqu'à zéro, en ôtant 1 chaque seconde :

```
10 FOR S=60 TO 1 STEP -1
20 CLS
30 PRINT S
40 WAIT 95
50 NEXT S
60 PRINT S
70 PING
```

REMARQUES

40 — pause pour battre la seconde

70 — le gong !

REPONSES AUX QUESTIONS SUR LE CHAPITRE V

1. Non, ça ne va pas ! Le nom de la variable ne doit être écrit qu'*une seule fois*. Il fallait écrire :

FOR N = 1 TO 23

2. Cette ligne est parfaitement correcte.

3. Celle-ci aussi.

4. Oui, celle-ci aussi : le pas est très grand, aussi n'y a-t-il que le 37 qui sera pris en compte car, au tour suivant, on passerait à 137 ce qui excède la limite de 73.

5. C'est parfaitement correct : le pas peut être tout petit.

6. Alors là, rien ne va plus : essayez donc de *décompter* (– 25) en accroissant des valeurs de 37 à 73 !

7. Ici, tout va bien car, avec le pas négatif, on a inscrit un décomptage de 37 à zéro.

8. C'est parfaitement correct. Rappelez-vous que le double point est un séparateur d'ordres sur une même ligne.

La suite des réponses page 96

8

DE ZÉRO À 99
FOR NEXT

Dresser le tableau des nombres de 0 à 99 en les groupant par 10 sur chaque ligne : 0 à 9 sur la première, 10 à 19 sur la deuxième... Pour cela on peut distinguer le chiffre des unités et celui des dizaines.

Pour chaque dizaine, il faudra donner au chiffre des unités toutes les valeurs de 0 à 9 et les afficher sur la même ligne. Puis on passera à la ligne et on répètera ceci pour que le chiffre des dizaines aille lui aussi de 0 à 9 :

```
10 CLS
20 FOR D=0 TO 9
30 FOR U=0 TO 9
40 PRINT U+10*D;
50 NEXT U
60 PRINT
70 NEXT D
```

REMARQUES

20 à 70 — boucle sur les dizaines

30 à 50 — boucle sur les unités

60 — à la ligne !

Lancez ce programme et vous obtiendrez :

```
0 1 2 3 4 5 6 7 8 9
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49
50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69
70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89
90 91 92 93 94 95 96 97 98 99
```

9

LE HASARD CET INCONNU
FOR RND INT

Pour vous permettre de bien exploiter la fonction RND qui fournit dans certains cas un nombre à peu près imprévisible, nous vous proposons ici quelques petites expériences sans danger, tant que vous n'engagez pas de pari !

D'abord un galop d'essai : tirage de 10 nombres grâce à RND(1) :

```
10 FOR N=1 TO 10
20 PRINT RND(1)
30 NEXT N
```

si vous utilisez ce programme plusieurs fois de suite, vous obtiendrez chaque fois 10 valeurs différentes :

```
.973136996
.103117626
.0177148333
.779343355
.551834438
.617419111
.960296981
.547150891
.802192734
.814107273
```

Après chaque utilisation du programme, tapez en mode immédiat :

```
PRINT RND(0)
.814107273
```

vous devez obtenir la dernière valeur fournie par le programme. Ça peut servir si on a besoin de cette valeur et qu'on ne l'a pas mise en conserve.

Et RND(– 1), par exemple ? Pendant la période de mise au point d'un programme, on est amené à changer une ligne par ci, une valeur par là. Si en plus le programme produit des valeurs aléatoires qui changent d'un essai au suivant, on ne peut plus rien contrôler. Avec une valeur négative dans sa parenthèse, la fonction RND fixe le point de départ d'une suite de valeurs obtenues avec RND (nombre positif).

```
10 PRINT"AMORCE ";RND(-1)
20 FOR N=1 TO 10
30 PRINT RND(1)
40 NEXT N
```

deux utilisations successives de ce programme fournissant la même suite de 10 valeurs, avec en-tête l'amorce.

```
RUN
AMORCE 2.99196472E-08
.738207502
.272707136
.299733446
.54034268
.878488765
.0517244711
.066466605
.893639669
.921827414
.535536507

RUN
AMORCE 2.99196472E-08
.738207502
.272707136
.299733446
.54034268
.878488765
.0517244711
.066466605
.893639669
.921827414
.535536507
```

En changeant la valeur négative on change l'amorce, donc la suite de valeurs.

```
10 PRINT"AMORCE ";RND(-2)
```

```
RUN
AMORCE 2.99205567E-08
.273385388
.621966945
.369081192
.300018938
.572401276
```

```
.048888672
.971839163
.434229892
.302435673
.458152464
```

Par contre si l'on change la valeur positive à la ligne 30, on ne change pas de suite.

```
30 PRINT RND(2)
```

```
RUN
AMORCE 2.99205567E-08
.273385388
.621966945
.369081192
.300018938
.572401276
.048888672
.971839163
.434229892
.302435673
.458152464
```

Et même si cette valeur change d'un tirage à l'autre, la suite reste immuable, derrière son chef de file : l'amorce.

```
30 PRINT RND(N)
```

```
RUN
AMORCE 2.99205567E-08
.273385388
.621966945
.369081192
.300018938
.572401276
.048888672
.971839163
.434229892
.302435673
.458152464
```

Nous pourrions ne pas fixer l'amorce afin de laisser le hasard jouer son rôle dès le début. Mais alors la première valeur tirée après la mise en marche de l'ordinateur sera toujours la même.

On obtiendra donc deux fois la même suite si on utilise deux fois le même programme, et que chaque fois aucun tirage n'a eu lieu depuis le branchement de l'ordinateur. Pour éviter cela, le programme va demander à l'utilisateur de penser à un nombre de 1 à 255 et surtout de le taper ! C'est ce nombre qui va servir de point de départ :

```
10 INPUT"TAPEZ UN NOMBRE DE 1 A 255";A
20 PRINT"AMORCE ";RND(-A)
30 FOR N=1 TO 10
40 PRINT INT(RND(1)*10)
50 NEXT N
```

Pour changer un peu, la ligne 40 va afficher des nombres entiers pouvant aller de 0 à 9 :

```
RUN
TAPEZ UN NOMBRE DE 1 A 255? 123
AMORCE 5.73991201E-08
3
0
9
4
5
3
3
6
1
7
```

Ici le 3 est sorti...3 fois !

Par curiosité, comptons le nombre de sorties, pour chaque numéro en rangeant ces nombres dans diverses cases d'un fichier que nous appellerons S. Une case est repérée par la valeur de ce qui est dans la parenthèse : T.

```
10 INPUT"TAPEZ UN NOMBRE DE 1 A 255";A
20 PRINT"AMORCE ";RND(-A)
30 FOR N=1 TO 10
35 T=INT(RND(1)*10)
40 PRINT T,
45 S(T)=S(T)+1
```

Ceci veut dire que si T vaut 2, on ajoute 1 dans la case n° 2 de S. Attention, la boucle n'est pas terminée.

```
50 NEXT N
55 :
60 PRINT:PRINT"BILAN DES TIRAGES"
70 FOR T=0 TO 9
80 PRINT"LE NUMERO ";T;" EST SORTI ";S(T
);" FOIS"
90 NEXT T
```

```
RUN
TAPEZ UN NOMBRE DE 1 A 255? 12
AMORCE 4.48235369E-08
7    3    5    6    7    7    5    9
  3    3
BILAN DES TIRAGES
LE NUMERO 0  EST SORTI 0  FOIS
LE NUMERO 1  EST SORTI 0  FOIS
LE NUMERO 2  EST SORTI 0  FOIS
LE NUMERO 3  EST SORTI 3  FOIS
LE NUMERO 4  EST SORTI 0  FOIS
LE NUMERO 5  EST SORTI 2  FOIS
LE NUMERO 6  EST SORTI 1  FOIS
LE NUMERO 7  EST SORTI 3  FOIS
LE NUMERO 8  EST SORTI 0  FOIS
LE NUMERO 9  EST SORTI 1  FOIS
```

Et pour un nombre variable de tirages :

```
20 PRINT"AMORCE ";RND(-A)
25 INPUT"NOMBRE DE TIRAGES";NB
30 FOR N=1 TO NB
35 T=INT(RND(1)*10)
```

Essayez ce programme pour un nombre de tirages plus ou moins importants. Tous les numéros ne sortent pas aussi souvent. Que se passe-t-il quand on augmente le nombre de tirages ?

NB : La notation S(T) sous-entend en principe une réservation de mémoire que nous n'avons pas faite ici parce que les conditions le permettaient. Pour être en règle il faudrait faire cette réservation avec l'instruction DIM que nous rencontrerons plus loin.

10

POUR RÉVEILLER, SONS ET COULEURS !

PING ZAP PAPER INK

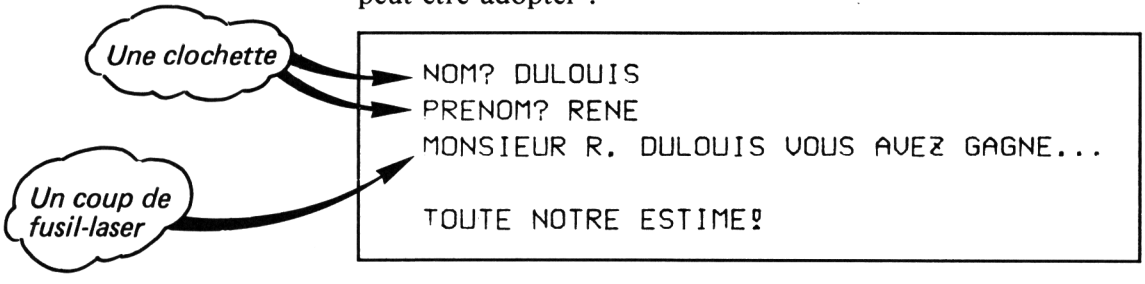
Lorsqu'un programme demande des données ou affiche des résultats de temps en temps, il peut être utile de le signaler au spectateur grâce à un son ou une manifestation visuelle. On peut par exemple sonner avant chaque instruction INPUT, et tirer un coup de laser dès qu'un résultat s'affiche.

```
10 CLS
20 PING:INPUT"NOM";N$
30 PING:INPUT~PRENOM";P$
40 IP$=LEFT$(P$,1)+". "
50 ZAP:PRINT"MONSIEUR ";IP$;N$;" VOUS AV
EZ GAGNE..."
60 PRINT"TOUTE NOTRE ESTIME!"
```

REMARQUES

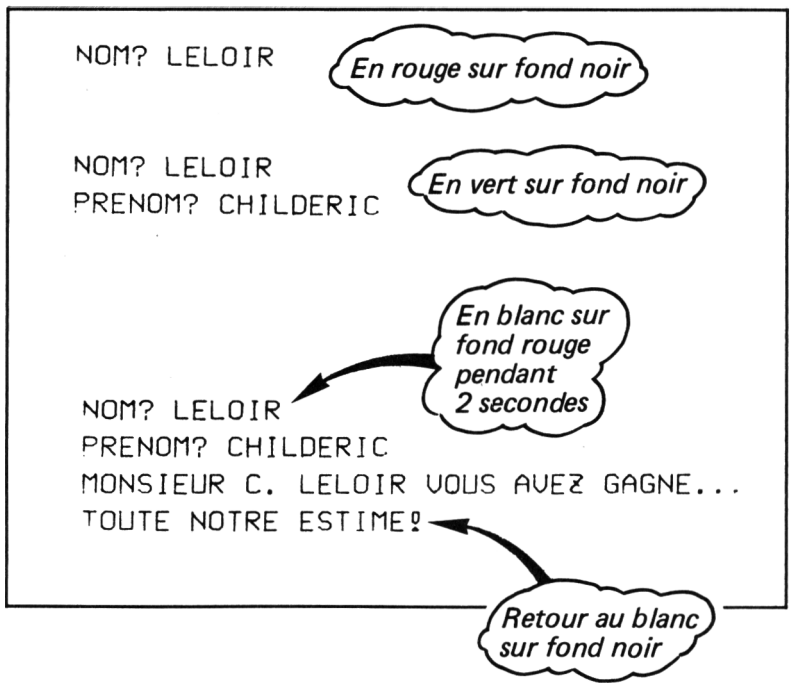
40 — on conserve l'initiale du prénom à laquelle on ajoute le point et un espace

Nous ne pourrons pas reproduire les effets sonores ici, essayer... c'est peut-être adopter !



Lorsque certains traitements sont longs, cette méthode est bien utile car elle évite de fixer l'écran dans l'attente de résultats. Mais si le son gêne (après 22 h !) vous pouvez le remplacer par de la couleur.

```
5 PAPER 0:INK 7
10 CLS
20 INK 1:INPUT"NOM";N$
30 INK 2:INPUT"PRENOM";P$
35 INK 7
40 IP$=LEFT$(P$,1)+". "
50 PAPER 1:PRINT"MONSIEUR ";IP$;N$;" VOU
S AVEZ GAGNE..."
52 WAIT 200
55 PAPER 0
60 PRINT"TOUTE NOTRE ESTIME!"
```



Si vous avez essayé ce programme, vous avez remarqué que les couleurs ne duraient que le temps de l'introduction ou de l'affichage : nous n'obtenons pas pour l'instant plusieurs couleurs d'encre ou de papier différents en même temps sur l'écran. Rendez-vous au chapitre suivant.

SUITE DES REPONSES AUX QUESTIONS SUR LE CHAPITRE V

9. Mais non, il y a une confusion entre la variable N et T qui sert à la définir. Il aurait fallu écrire, apparemment :

FOR N=T+2 TO T+5:?N: NEXT N

10. PAPER 3 fait passer la page en jaune.

11. Par contre INK = 1 fournit une erreur de syntaxe ! Il ne faut pas mettre de signe *égal ;* pour faire passer le texte en rouge, il suffit (et il faut) écrire INK 1.

12. Rien du tout ! Lorsqu'il rencontre un ordre REM, l'ordinateur saute la suite et passe à la ligne suivante.

13. L'ordre RND sert à tirer une valeur aléatoire.

14. Oui : RND(9) et RND(18) fournissent le même résultat. La valeur du nombre entre parenthèses est sans aucune importance pour autant qu'il soit positif et non nul : il est, en quelque sorte, fictif.

15. Non, RND(6) fournit un nombre aléatoire compris entre Ø et 1 et ne convient donc pas. Réfléchissez un instant sur la seconde formule :
RND(9) fournit une valeur entre Ø et 1 exclu
RND(9)*6 donne une valeur entre Ø et 6 exclu
INT(RND(9)*6) donne un entier compris entre Ø et 6 exclu, donc entre Ø et 5 inclus
INT(RND(9)*6)+1 donne un entier entre 1 et 6 inclus, ce que vous voulions obtenir.

A vous, maintenant d'imaginer un programme pour jouer au 421 !

11

NOTATION SCIENTIFIQUE

1.23456789E + 10

Si un nombre est très petit, ou très grand, on préfère souvent l'écrire comme s'il était voisin de 1 en corrigeant le tir grâce à un multiple (ou un sous-multiple) de 10 :

0.0000001 s'écrit ainsi 1E – 7

c'est-à-dire une fois dix à la puissance moins sept. Pour vous familiariser avec ces petits et ces grands, nous allons regarder dans quel cas la notation scientifique intervient, grâce à un programme qui demande un nombre et qui le multiplie et le divise par 10, 100, 1000... jusqu'à ce que l'ordinateur ne puisse plus représenter un des résultats dans ses mémoires : « OVERFLOW ERROR IN 60 ».

```
10 CLS
20 INPUT"UN NOMBRE SVP";N
30 N1=N*10
40 N2=N/10
50 PRINT N1,N2
60 N1=N1*10
70 N2=N2/10
80 GOTO 50
```

Si on tape par exemple le nombre 185.637 on observe un déplacement du point (notre virgule) vers la droite lorsqu'on multiplie et vers la gauche lorsqu'on divise. Tant mieux ! Puis apparaît un E dans la colonne de gauche et un peu plus tard dans celle de droite, pour éviter d'accumuler trop de zéros à droite ou à gauche.

```
UN NOMBRE SVP? 185.637
1856.37     18.5637
18563.7     1.85637
185637      .185637
1856370     .0185637
18563700    1.85637E-03
185637000   1.85637E-04
1.85637E+09         1.85637E-05
1.85637E+10         1.85637E-06


1.85637E+36         1.85637E-32
1.85637E+37         1.85637E-33
?OVERFLOW ERROR IN 60
Ready
```

Avec 1 au départ, le programme se bloque au moment où N1 cherche à prendre la valeur 1E + 39 c'est-à-dire un nombre formé d'un 1 et de 39 zéros.

```
UN NOMBRE SUP? 1
10          .1
100         .01
1000        1E-03
10000       1E-04
100000      1E-05
1000000     1E-06
10000000    1E-07
100000000   1E-08
1E+09       1E-09
1E+10       1E-10

1E+35       1E-35
1E+36       1E-36
1E+37       1E-37
1E+38       1E-38
?OVERFLOW ERROR IN 60
Ready
```

Si vous essayez 0.01, le blocage a lieu au même endroit, et N2 est devenu nul.

Pourriez-vous trouver vers quelle valeur se produit le blocage, et quand un nombre trop petit est-il transformé en zéro ?

12

CONVERSIONS BINAIRE - DÉCIMALE
FOR INT

Vous avez vu au chapitre 1 qu'un octet représentait un nombre binaire composé de 8 chiffres, des zéros ou des uns. Les enfants de 6 ans évoluent en base deux comme un poisson dans l'eau. Mais c'est plutôt la base dix qui fait partie de la vie de tous les jours... heureusement !

Voici deux programmes permettant la conversion d'un nombre écrit dans notre bon vieux système décimal en un nombre binaire, et inversement. Ils sont faits pour des nombres compris entre 0 et 255, c'est-à-dire 0 et 11111111 en binaire : c'est ce que peut contenir un octet.

Conversion décimale — binaire

```
10 REM CONVERSION DECIMAL-->BINAIRE
20 INPUT"NOMBRE EN BASE DIX";N
30 B$=""
40 FOR P=0 TO 7
50 Q=INT(N/2)
60 R=N-Q*2
70 B$=STR$(R)+B$
80 N=Q
90 NEXT P
100 PRINT"EN BASE DEUX : ";B$
```

REMARQUES

30 — B$ contiendra les zéros et les uns : on commence par ne rien mettre

40 à 90 — boucle de recherche des 8 chiffres binaires

50 — quotient entier de la division de N par 2

60 — reste de cette division : c'est un des huit chiffres que l'on cherche

70 — ce chiffre vient se placer à gauche des précédents dans B$

80 — on va refaire la même chose avec Q...

90 — ... jusqu'à épuisement des valeurs de P

Exemples :

```
RUN
NOMBRE EN BASE DIX? 2
EN BASE DEUX : 00000010

Ready
RUN
NOMBRE EN BASE DIX? 255
EN BASE DEUX : 11111111

Ready
RUN
NOMBRE EN BASE DIX? 256
EN BASE DEUX : 00000000
```

```
Ready
RUN
NOMBRE EN BASE DIX? 257
EN BASE DEUX : 00000001

Ready
RUN
NOMBRE EN BASE DIX? -255
EN BASE DEUX : 00000001

Ready
```

Conversion binaire — décimale

```
10 REM CONVERSION BINAIRE-->DECIMAL
20 INPUT"NOMBRE EN BASE DEUX";B$
30 D=0
40 FOR P=0 TO 7
50 D=D+VAL(MID$(B$,8-P,1))*(2^P)
60 NEXT P
70 PRINT"EN BASE DIX : ";D
```

REMARQUES

20 — le nombre binaire est logé dans B$: il est considéré comme une chaîne de caractères

30 — on met à zéro ce qui contiendra le résultat

40 à 60 — boucle de constitution du nombre décimal

50 — si le chiffre binaire n° P vaut 1, on ajoute 2^P à D

Exemples :

```
RUN
NOMBRE EN BASE DEUX? 00000010
EN BASE DIX : 2

Ready
RUN
NOMBRE EN BASE DEUX? 01000001
EN BASE DIX : 65

Ready
```

CHAPITRE VI

L'ART GRAPHIQUE EN BASSE RÉSOLUTION

Comment utiliser l'écran pour des tracés, des dessins et même un début d'animation, c'est ce qu'on va étudier dans ce chapitre. Il est consacré aux modes graphiques dits à basse résolution, car l'écran est « divisé » en 39 colonnes sur 27 rangées. On passera au mode « haute résolution » dans le chapitre suivant, bien plus fin puisqu'alors, l'écran dispose de 240 par 200 « points » graphiques.

Principaux thèmes abordés

Mode texte	Etoiles
TEXT	RUN numéro de ligne
Modes graphiques	END
Grille graphique	STOP
PLOT	Sous-programme
Couleurs avec PLOT	Programme principal
Barres de couleur	GOSUB
Graphique basse résolution	RETURN
LORES	LORES 1
HIRES	Caractères graphiques
LORES Ø	CTRL/C
Traits en LORES Ø	CONT
Carré en LORES Ø	Mouvement
Duplication d'instructions	CHR$(8)
Suppression du curseur	CHR$(9)
CTRL/Q	

1. L'organisation de l'écran en « mode texte »

Pour écrire du texte sur l'écran, des lignes les unes à la suite des autres, on n'a pas eu à se préoccuper de son organisation. Vous savez, cependant, que la « page » de texte, le cadre blanc affiché, comporte 27 lignes et 40 colonnes, dont seules, 38 sont utilisables. La ligne supérieure, au-dessus de ce cadre (la 28e, donc), servait à divers messages.

MODE TEXTE
TEXT

Ce « mode » de travail que l'on a exploité, jusqu'à présent, s'appelle « le mode texte », car il nous servait à écrire... du texte. En anglais : TEXT.

Vous verrez l'importance de ce point quand on examinera les *trois* autres modes, dits *graphiques*.

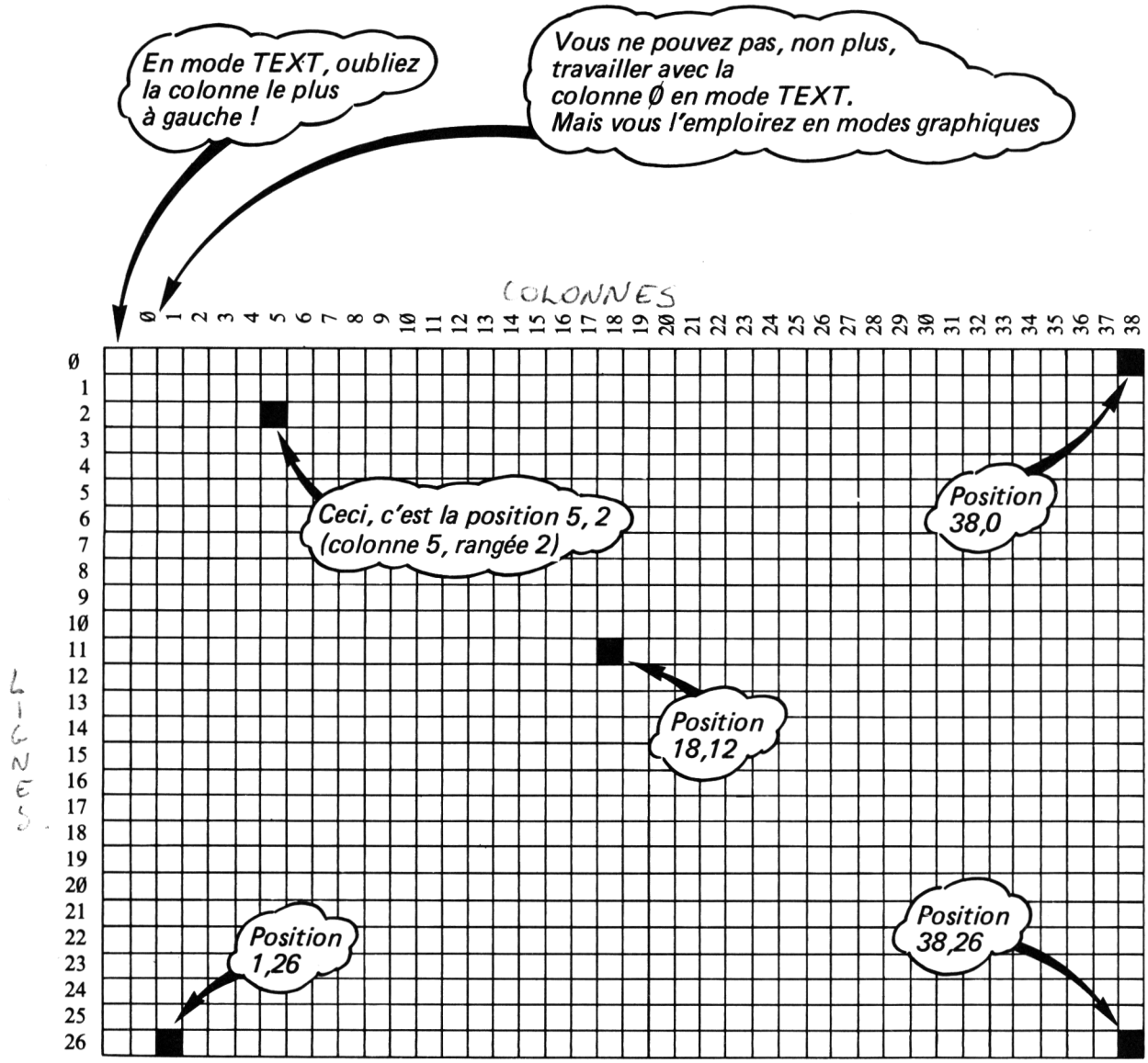
MODES GRAPHIQUES

C'est dans ce cadre utile de 27 par 4Ø que nous allons commencer par écrire, puis dessiner. Pour cela, Il nous faut désigner des positions précises ; on va dessiner cette page avec ses lignes et ses colonnes, puis on va numéroter les lignes *et* les colonnes (voyez notre dessin) :

• *Pour les lignes,* aucun problème : on va de Ø à 26 (ce qui fait bien 27 au total).

GRILLE GRAPHIQUE BASSE RESOLUTION

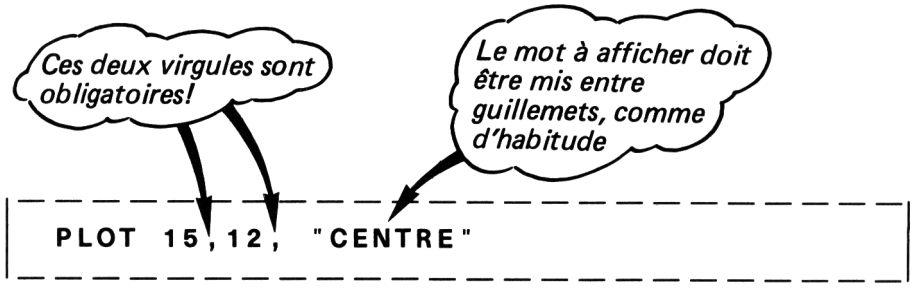
• *Pour les colonnes :* on commence par éliminer la colonne la plus à gauche, inutilisable, on l'a vu. Le numéro Ø sera donc attribué à la seconde colonne, que l'on ne pourra pas plus utiliser. Patience ! Puis, on ira jusqu'à 38.



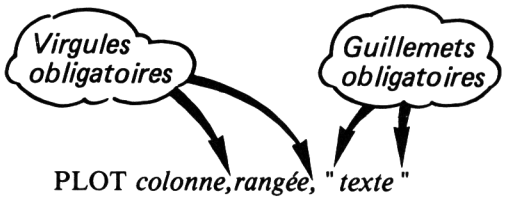
La grille d'écran en mode « texte ».

PLOT

Voulez-vous écrire un texte *au centre de l'écran,* ligne 12, par exemple ? N'employez plus PRINT, mais PLOT (pour « dessiner »), en indiquant la position *colonne, rangée,* puis le texte à afficher. Supposons qu'on parte de la colonne 15 ; faites d'abord CTRL/L pour effacer l'écran, puis entrez en mode direct :



Avec le RETURN, le mot CENTRE s'inscrit... au centre de l'écran. Vérifiez sa position exacte sur notre grille. Faites quelques autres essais de votre crû, mais rappelez-vous que la « syntaxe » correcte de l'ordre est :



Voici donc une façon de loger du texte là où bon vous semble. Mais pourquoi ne pas user de la couleur ? On peut, en effet, compléter PLOT avec un « attribut » de couleur, comme l'on dit dans le jargon « oricien » : c'est un nombre de plus, précisant la couleur.

Avec PLOT et pour mettre en couleur :
• Le texte écrit (donc, l'équivalent de INK), utilisez les chiffres de couleurs de Ø à 7, comme indiqués précédemment.
• Le fond (la « page »), utilisez ces mêmes chiffres auxquels il faut ajouter 16.

Voici ce que cela donne :

COULEURS AVEC PLOT

Avec l'ordre PLOT

COULEUR	NUMERO ATTRIBUT	
	Pour l'écriture	Pour le fond
Noir	Ø	16
Rouge	1	17
Vert	2	18
Jaune	3	19
Bleu	4	20
Magenta	5	21
Cyan	6	22
Blanc	7	23

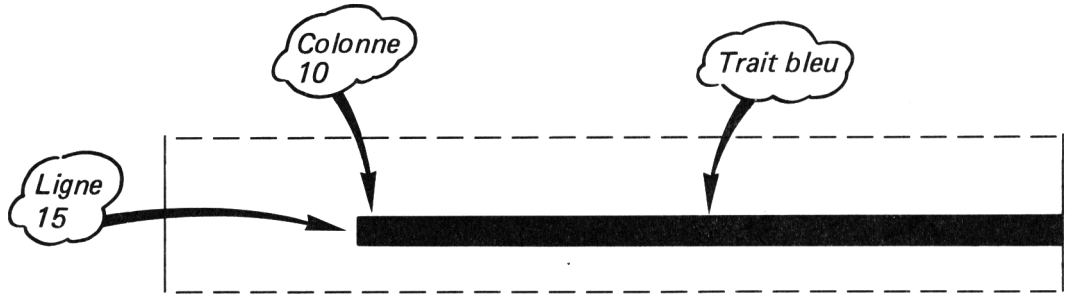
C'est toujours du bleu-vert

Maintenant, la syntaxe de l'ordre PLOT va être :

En effet, il faut choisir : l'attribut ou le texte, mais pas les deux à la fois avec PLOT. Essayons de passer en *bleu* (attribut 2Ø) la ligne 15, à partir de la colonne 1Ø. Commencez par faire CTRL/L, puis entrez, en mode direct :

```
PLOT 1Ø,15,2Ø
```

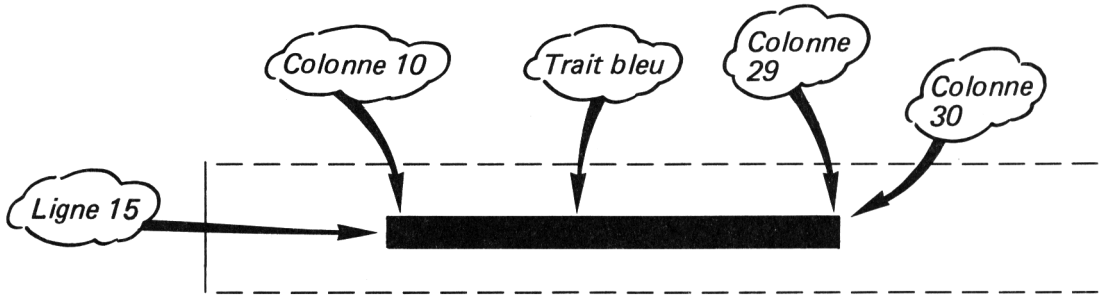
Au RETURN, toute la ligne 15, *à partir de* la colonne 1Ø, se teinte en bleu, ce qui donne (avec de l'imagination pour la couleur) :



Voulez-vous *effacer* cette couleur, de la colonne 3Ø à la fin ? Il suffit de re-colorier *en blanc* (attribut : 23) cette ligne, avec :

```
PLOT 3Ø,15,23
```

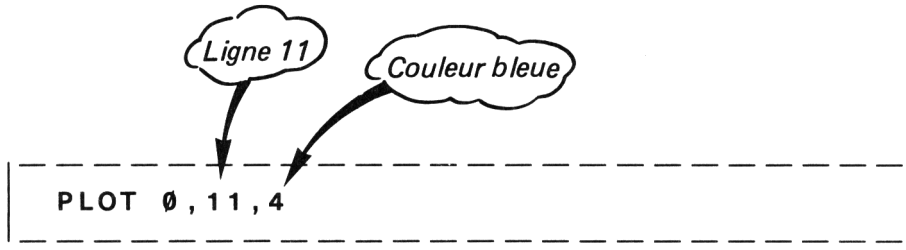
Le RETURN vous donnera :



Rien ne vous empêche de commander maintenant, toujours en mode direct, PAPER 1 pour voir la page devenir rouge, avec un trait bleu central, prolongé par du blanc.

A ce propos, remarquez qu'une nouvelle commande PAPER, ou même INK, ne modifie pas la couleur d'une position déjà coloriée.

La même structure d'ordre, employant comme attributs de couleur les chiffres de Ø à 7, teinterait le texte (c'est l'équivalent de INK) et non le fond, selon le même processus. Avec une petite différence : le caractère de la première cellule s'efface ! En effet, rappelez-vous qu'il faut choisir : la couleur ou le caractère. Vérifiez-le en commandant le passage en couleurs d'une ligne de texte apparaissant sur notre écran. Pour éviter cet effacement et colorier *toute* la ligne, partez de la colonne Ø. Ainsi, si votre texte se trouve sur la ligne 11 et que vous vouliez le mettre en bleu, frappez :



Vous verrez *toute* la ligne 11 écrite en bleu.
Mais faisons mieux : on va afficher des lignes de couleurs complètes, en choisissant la ligne *au hasard* et la couleur *au hasard ;* parce que le programme va boucler sans fin, une nouvelle couleur chassera l'autre dans un effet global très coloré et esthétiquement déjà discutable.

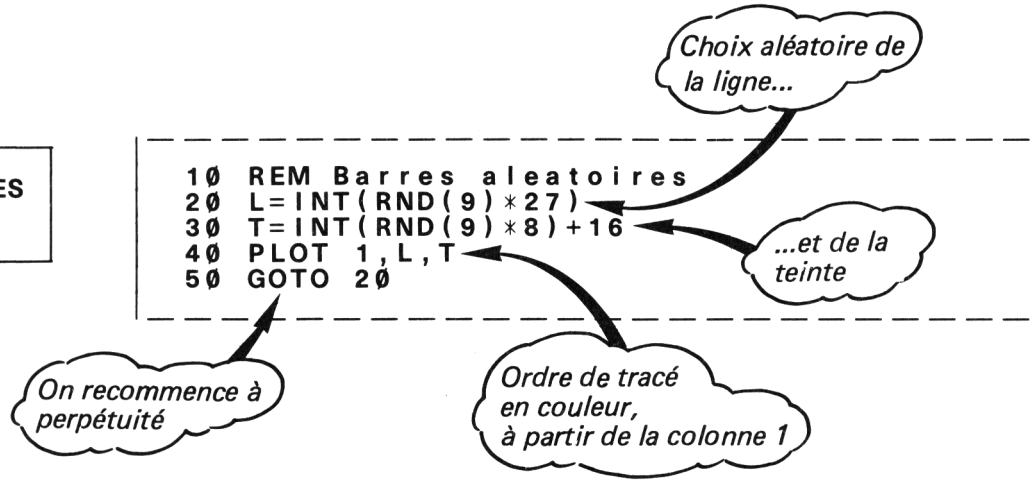
Il nous faut d'abord choisir une ligne (appelons-la L) au hasard, mais obtenir une valeur entre Ø et 26 *inclus.* Rappelez-vous que RND(9) fournit une valeur comprise entre Ø et 1 *exclus,* donc entre Ø et, dans notre écriture habituelle : Ø,99999... Il faudra donc multiplier la valeur aléatoire par 27 puis en extraire l'entier, ce qui s'écrira INT(RND(9)*27).

Rappelez-vous aussi que le premier (9) entre parenthèses n'a *aucune* signification ; vous pouvez mettre 8 ou 33 ou 456, cela ne changera rien tant qu'il ne s'agira pas d'une valeur nulle ou négative.
Pour la couleur de la ligne, qu'on va appeler T, pour *teinte,* c'est un peu plus compliqué : il faut obtenir des valeurs de 16 à 23 incluses, donc de Ø + 16 à 7 + 16 incluses. Qu'à cela ne tienne : de Ø à 7, nous savons le faire ; il suffira d'ajouter 16, ce qui donnera l'ordre :

$$T = INT(RND(9) * 8) + 16$$

Dès lors, le programme est simple à rédiger :

BARRES ALEATOIRES DE COULEUR



Lancez ce programme, et vous verrez des lignes colorées remplir l'écran et se remplacer au hasard, pour les lignes et pour les couleurs.

2. Graphique basse résolution, mode Ø

GRAPHIQUE BASSE RESOLUTION

Abordons, maintenant, le mode graphique. Il en existe deux :

• L'un est appelé le mode graphique « *basse résolution* », ce qui s'écrit LORES, de l'anglais « *low resolution* », c'est-à-dire basse résolution. C'est par lui que nous allons commencer.

• L'autre est le mode graphique « *haute résolution* », qui s'écrit HIRES, de « *high resolution* ».

LORES
HIRES

Le mode LORES se divise à son tour en deux modes, appelés tout bonnement LORES Ø et LORES 1. A nouveau, nous allons d'abord examiner le premier, LORES Ø.

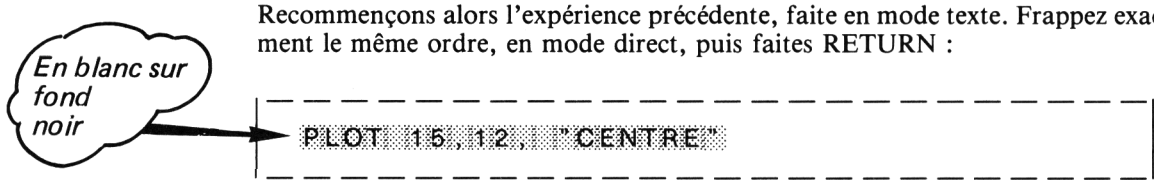
Commençons par informer l'Oric qu'on passe en mode LORES Ø ; pour cela, frappez en mode direct :

LORES Ø

```
LORES Ø
```

Puis faites RETURN. Instantanément, l'écran devient noir, car c'est ainsi que fonctionne LORES Ø ; tout le texte en noir qui pré-existait disparaît du même coup, alors qu'un *Ready* en blanc et un curseur blanc aussi se placent là où il fallait les attendre, à la suite du texte pré-existant disparu.

Recommençons alors l'expérience précédente, faite en mode texte. Frappez exactement le même ordre, en mode direct, puis faites RETURN :

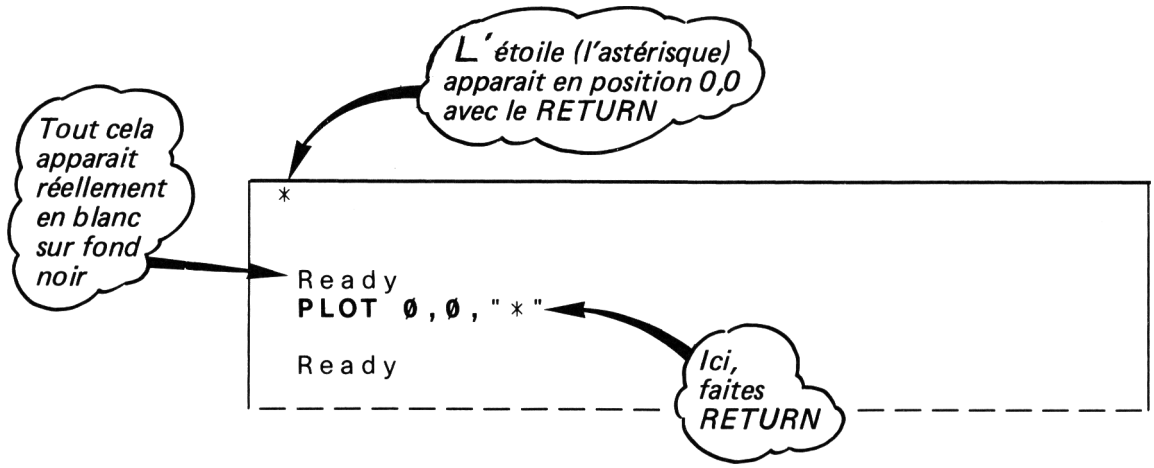


Nous ne pouvons utilement reproduire ici tout l'écran, noir... Au RETURN, le mot CENTRE s'affiche au centre de l'écran, mais en blanc, cette fois.

En « mode direct », LORES Ø (ou LORES 1) ne prolongent leur action que pour la seule « page » affichée. Si vous écrivez au-delà des 27 lignes, une « page blanche », en mode TEXT, ré-apparaît.

En mode LORES Ø (ou LORES 1), vous pouvez vous servir de la colonne Ø.

Vérifiez-le : *faites d'abord CTRL/L pour effacer l'écran,* ce qui vous ramène au mode TEXT puisque l'écran redevient blanc, *puis frappez LORES Ø,* et enfin un caractère quelconque, par exemple l'astérisque (mais placé entre guillemets). Cela donne :



A propos : rappelez-vous du code ASCII ; l'astérisque a pour code 42, ce qui fait que vous auriez aussi bien pu écrire CHR$(42) pour obtenir le même résultat. Vérifiez-le, en modifiant la position, par exemple :

```
*

Ready
PLOT Ø,Ø,"*"

Ready
PLOT 24,Ø,CHR$(42)

Ready
```

Cet ordre place l'étoile ici

Notons, maintenant, un point essentiel relatif à la couleur :

En mode LORES Ø (comme avec le mode LORES 1 qu'on va examiner ensuite), le placement de la couleur se fait position par position.

Cela veut dire qu'appliquer de la couleur sur une position ne remplira pas le reste de la ligne. Ainsi, frappez CTRL/L pour effacer la page, puis entrez en mode direct :

Rappelez-vous que le double-point sert à séparer deux instructions sur une même ligne

Couleur rouge, pour le fond

```
LORES Ø : PLOT 1Ø,5,17
```

Avec le RETURN, la page devient noire et, sur ce fond, vous obtiendrez un *unique* pavé rouge :

Le curseur : en blanc !

Pavé rouge en position 10,5

```
Ready
```

Le fond est noir!

Expérimentons alors le mode LORES Ø en tirant un trait horizontal sur la ligne 5, en rouge (couleur de page 17), d'un bout à l'autre de l'écran (de la colonne Ø à la 38). Il faudra placer tous les pavés successivement, à partir de la position Ø,5 jusqu'à 38,5 ce qui donne (faites un NEW, au préalable) :

```
NEW

10 REM Trait horizontal
20 CLS
30 LORES 0
40 FOR N=0 TO 38
50 PLOT N,5,17
60 NEXT N

RUN
```

Pour effacer l'écran

Au RETURN, un beau trait horizontal rouge apparaît sur fond noir :

Ready et le curseur sont en blanc

Ready

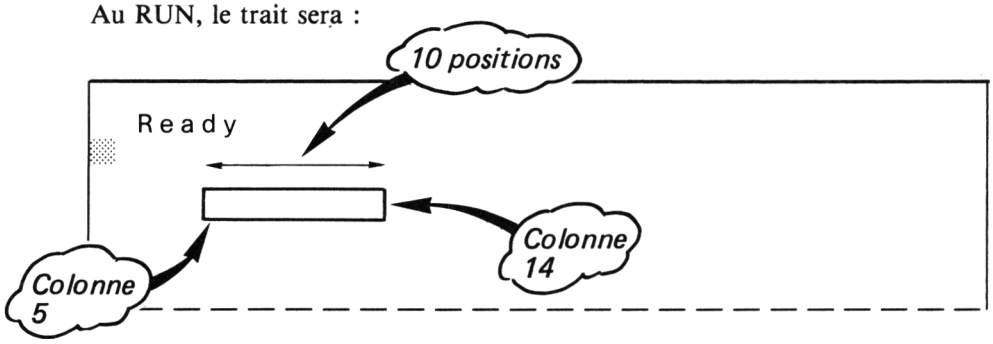
Trait rouge sur la ligne 5

TRAITS EN LORES 0

Pour obtenir un trait allant de la colonne 5 à la colonne 14, donc sur 10 positions, modifiez la ligne 40 ainsi :

```
40 FOR N=5 TO 14
```

Au RUN, le trait sera :



Pour un trait vertical en *colonne* 5, de 10 positions, le principe est exactement le même et le programme serait (ne modifiez que la ligne 50) :

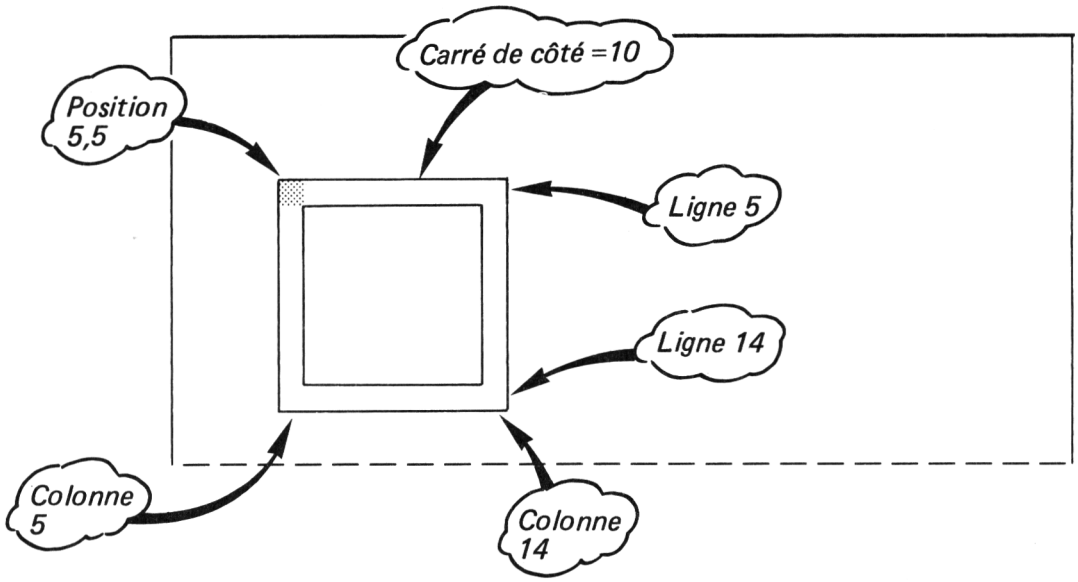
Changez aussi le titre!

```
10 REM Trait vertical
20 CLS
30 LORES 0
40 FOR N=5 TO 14
50 PLOT 5,N,17
60 NEXT N
```

On a simplement déplacé N, pour viser les rangées successives

Au RUN, vous verrez un trait vertical rouge s'installer sur la colonne 5, et sur 10 positions verticales.

De là à créer un carré complet, il n'y a qu'un pas. Voici ce qu'on veut obtenir, à partir des mêmes positions :



Et voici le programme correspondant :

CARRE EN LORES Ø

```
10 REM Carré
20 CLS
30 LORES Ø
40 FOR N=5 TO 14
50 PLOT N,5,17
60 PLOT N,14,17
70 PLOT 5,N,17
80 PLOT 14,N,17
90 NEXT N
```

Les deux côtés horizontaux

Les deux côtés verticaux

Faites un RUN : vous obtiendrez le résultat escompté (heureusement !). Mais ce fond noir n'est pas heureux : pour passer le fond en vert (couleur 18), par exemple, il faut badigeonner successivement tous les pavés, selon le même principe, colonne par colonne puis, et pour chaque colonne, rangée par rangée. Pour ce faire, on va employer deux boucles. Voici ce programme ; commencez-le à la ligne 1ØØ, par exemple, sans faire NEW :

C pour «colonne» et R pour »rangée»

```
100 REM Fond en vert
110 FOR C=Ø TO 38
120 FOR R=Ø TO 26
130 PLOT C,R,18
140 NEXT R
150 NEXT C
```

Voici les 2 boucles

Lancez l'ensemble avec RUN : le carré rouge apparaît, puis tout l'écran est passé au vert et le carré disparaît ! Pour bien faire, il faut faire passer les anciennes instructions 4Ø à 9Ø *après* 15Ø.

DUPLICATION D'INSTRUCTIONS

Pour déplacer des instructions, on peut les dupliquer. Listez le programme puis, comme si vous vouliez corriger une ligne, modifiez leur nouveau numéro en le frappant. Sortez de la ligne avec CTRL/A, puis faites RETURN. La ligne modifiée est dupliquée : dans un nouveau listage, elle apparaît deux fois : avec l'ancien et avec le nouveau numéro de ligne.

Il ne reste plus, alors, qu'à supprimer les lignes inutiles. Si vous faites cela, vous devez obtenir (en reprenant les numéros suivants) :

```
10 REM Carré
20 CLS
30 LORES 0

110 FOR C=0 TO 38
120 FOR R=0 TO 26
130 PLOT C,R,18
140 NEXT R
150 NEXT C

160 FOR N=5 TO 14
170 PLOT N,5,17
180 PLOT N,14,17
190 PLOT 5,N,17
200 PLOT 14,N,17
210 NEXT N
```

On a supprimé de 40 à 100

Pour remplir l'écran de vert : comme ci-dessus

Pour afficher le carré rouge : instructions 40 à 90 de la page précédente dupliquées à ces nouvelles lignes

Le RUN va verdir l'écran, puis tracer le carré rouge précédent.

3. Ciel d'étoiles

Puisqu'on peut placer des astérisques (des « étoiles ») sur l'écran, pourquoi ne pas essayer de le remplir de façon aléatoire ? On va accrocher des étoiles sur les positions R (rangée) et C (colonne) choisies au hasard avec RND :

```
10 REM Des étoiles
20 LORES 0
30 C=INT(RND(9)*39)
40 R=INT(RND(9)*27)
50 PLOT C,R, CHR$(42)
60 GOTO 30
```

Pour passer au mode LORES 0

La colonne : un nombre aléatoire, calculé entre 0 et 38 inclus

La rangée : un nombre aléatoire, calculé entre 0 et 26 inclus

Rappelez-vous le mode de calcul employé pour le tir galactique, et vérifiez qu'avec la ligne 30, on obtiendra bien tous les entiers entre 0 et 38 *inclus* alors qu'avec la ligne 40, on ira de 0 à 26 *inclus*.

SUPPRESSION DU CURSEUR

CTRL/Q

Faites un RUN : l'écran devient noir, et des étoiles vont s'allumer au hasard, jusqu'à remplir totalement l'écran : nous ne vous le dessinons pas !
Au cours de l'exécution de ce programme, le curseur blanc s'entête à rester sur sa ligne et continue à clignoter, ce qui gâche un peu l'effet. Or, on peut le supprimer. Stoppez ce programme avec CTRL/C, puis faites CTRL/L pour revenir à un écran blanc.

Frappez alors sur CTRL/Q : le curseur disparaît et restera absent tant que vous n'aurez pas frappé à nouveau CTRL/Q.

Or, CTRL/Q dispose d'un code numérique, qui est 17. On peut le programmer avec CHR$(17) ; ajoutez alors au début du programme :

```
15 ? CHR$(17)
```

Si vous refaites un RUN, vous constaterez que le curseur a disparu. *Attention, toutefois :* après le CTRL/C qui stoppe ce programme, il ne revient pas de lui-même (rappelez-vous que l'ordinateur se borne à obéir aux ordres) ; aussi vous faudra-t-il frapper CTRL/Q, ou donner l'ordre, en mode direct : ? CHR$(17).
Fort bien, mais des étoiles blanches sur un ciel noir, c'est peu poétique. Pourquoi pas des étoiles sur un ciel bleu (couleur n° 4), par exemple ? On a vu, précédemment comment procéder, alors allons un peu loin : on va aussi effacer des étoiles, aléatoirement, avec RND. Pour effacer une étoile sur un ciel bleu, il suffira de placer un pavé bleu à sa place (aléatoirement). L'ordre sera donc PLOT C,R,2Ø. Ajoutons donc cet ordre en mêlant ces programmes, ce qui obligera à refrapper la suite afin d'obtenir le programme complet suivant, qui démarre en 1ØØ. Notez encore deux choses :

1) — On a supprimé l'ordre INT avec RND. Maintenant que vous savez à quoi il sert, on peut préciser que, dans le cas précis de PLOT, ce dernier prend de lui-même la valeur entière des nombres qu'on lui propose !

2) — On a réservé un cadre bleu tout autour du ciel. Donc, les colonnes iront de 1 à 37 et non de Ø à 38, et les rangées de 1 à 25 et non de Ø à 26.

ETOILES DANS LE CIEL

Voici donc le programme au complet qu'on a *volontairement* démarré en 1ØØ (surtout, ne faites pas NEW si vous voulez qu'on puisse vous faire une autre démonstration) :

```
1ØØ REM Quel beau ciel !
11Ø ? CHR$(17)
12Ø LORES Ø
13Ø FOR C=Ø TO 38
14Ø FOR R=Ø TO 26
15Ø PLOT C,R,2Ø
16Ø NEXT R
17Ø NEXT C
18Ø C=RND(9)*37+1
19Ø R=RND(9)*25+1
2ØØ PLOT C,R,"*"
21Ø C=RND(9)*37+1
22Ø R=RND(9)*25+1
23Ø PLOT C,R,2Ø
24Ø GOTO 18Ø
```

Séquence pour mettre le ciel en bleu

Séquence pour allumer les étoiles

Séquence pour éteindre des étoiles

RUN NUMERO DE LIGNE

Pour lancer ce programme, ne faites pas RUN seulement car ce serait le programme précédent qui démarrerait, mais précisez la ligne : RUN 1ØØ. Dès lors, le Basic ira directement à 1ØØ, ignorant ce qui précède.

On peut lancer n'importe quel programme en mémoire (lorsqu'il y en a plusieurs) en indiquant son numéro de ligne. En l'absence d'un numéro de ligne, le Basic recherche le programme qui commence par le plus faible numéro de ligne.

Donc, faites :

```
RUN 1ØØ
```

Vous verrez alors des étoiles s'allumer ou s'éteindre sur un fond de ciel bleu. Après le CTRL/C qui stoppera ce programme, n'oubliez pas de faire CTRL/Q pour rétablir le curseur.

END

Puisque nous vous avons montré comment démarrer un programme à partir de n'importe quelle ligne, ajoutons qu'on peut également marquer une fin obligatoire. Ce sera utile lorsque plusieurs programmes se succèdent. A la fin d'un programme, vous placerez l'ordre END (« fin »), comme au cinéma, avec un numéro d'instruction. Par exemple :

```
625 END
```

STOP

Le Basic, rencontrant END, s'arrêtera et n'ira pas chercher des instructions ultérieures qui existeraient. Ou encore, on pourra utiliser l'ordre STOP : lorsque le BASIC le rencontre, il cesse l'exécution mais affiche le numéro de ligne sur lequel il s'est arrêté.

4. Sous-programmes

Vous vous êtes familiarisé avec le maniement de la basse résolution, en LORES Ø avec le programme précédent. Celui-ci comporte cependant une redondance peu élégante : les lignes 18Ø et 19Ø sont les mêmes que les lignes 21Ø et 22Ø. Voici comment supprimer ce phénomène.

On va « extraire » ces deux lignes et les placer ailleurs, par exemple en 5ØØ. Quand on en aura besoin, on fera appel à elles.

SOUS-PROGRAMME

PROGRAMME PRINCIPAL

Cette séquence distincte à laquelle on peut faire appel à volonté est désignée par le nom de « sous-programme ».
Par opposition, le programme appelant est le « programme principal ».

A la fin du sous-programme, on sera néanmoins contraint d'ajouter une instruction précisant que le sous-programme est terminé et qu'il faut revenir au programme principal.

GOSUB
RETURN

De ce fait, deux ordres nouveaux interviennent :

• GOSUB, qui « appelle » le sous-programme

et est situé dans le programme principal.

• RETURN qui renvoie au programme principal après exécution du sous-programme. Le renvoi se fait toujours et automatiquement à l'instruction qui suit l'appel par GOSUB.

Le plus simple consiste à vous montrer comment transformer le programme précédent : supprimez les lignes 19Ø et 22Ø, modifiez 18Ø et 21Ø, et écrivez les nouvelles lignes 5ØØ à 53Ø comme suit (la REM est en prime !).

Attention : il faut écrire RETURN et non frapper sur la touche du même nom, à la ligne 53Ø. C'est là une source de confusion regrettable !

Voici donc ce programme avec son sous-programme :

```
100 REM Quel beau ciel !
110 ? CHR$(17)
120 LORES 0
130 FOR C=0 to 38
140 FOR R=0 TO 26
150 PLOT C,R,20
160 NEXT R
170 NEXT C
180 GOSUB 500

200 PLOT C,R,"*"
210 GOSUB 500

230 PLOT C,R,20
240 GOTO 180


500 REM Sous-programme
510 C=RND(9)*37+1
520 R=RND(9)*25+1
530 RETURN
```

1er appel du sous-programme

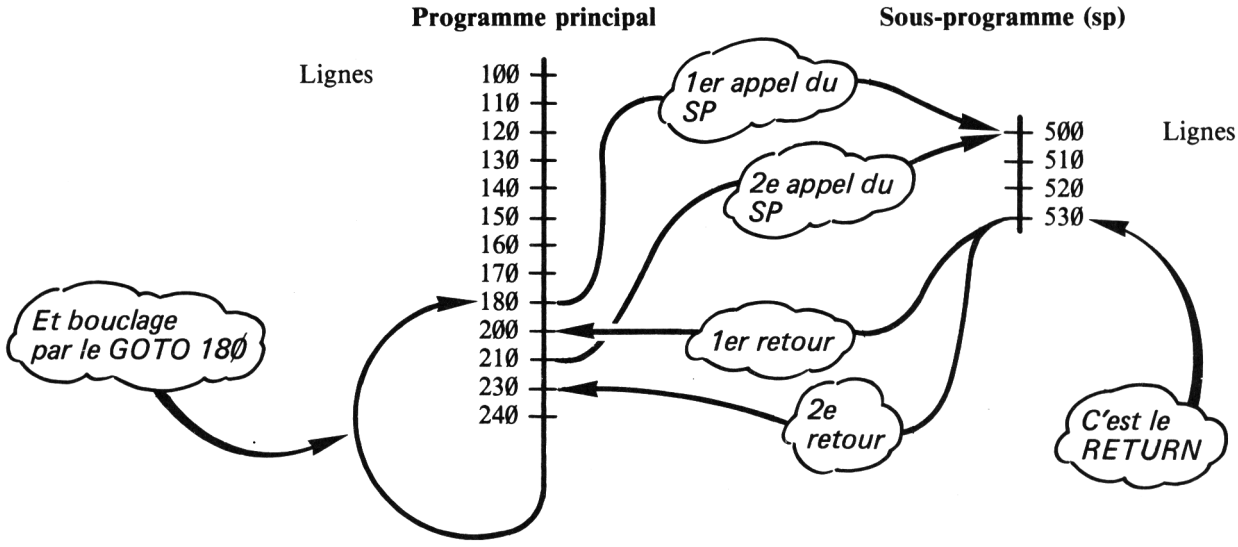
Ligne 190 supprimée

2e appel du sous-programme

Ligne 220 supprimée

Ordre de retour au programme principal

On peut illustrer ainsi ce programme :



Rappelez-vous que le retour se fait automatiquement à l'instruction qui suit celle d'appel, le « GOSUB ».

Les sous-programmes ne sont réellement très intéressant que lorsque la séquence passée en sous-programme est longue, ou lorsqu'elle doit intervenir de nombreuses fois. Ce pourrait être le cas si elle devait intervenir *dans des programmes différents.* Avec un sous-programme, en effet, on ne frappe qu'une seule fois la liste des instructions répétitives ! En outre, on économiserait de la mémoire centrale (mais pas dans notre cas, où le sous-programme était vraiment trop court).

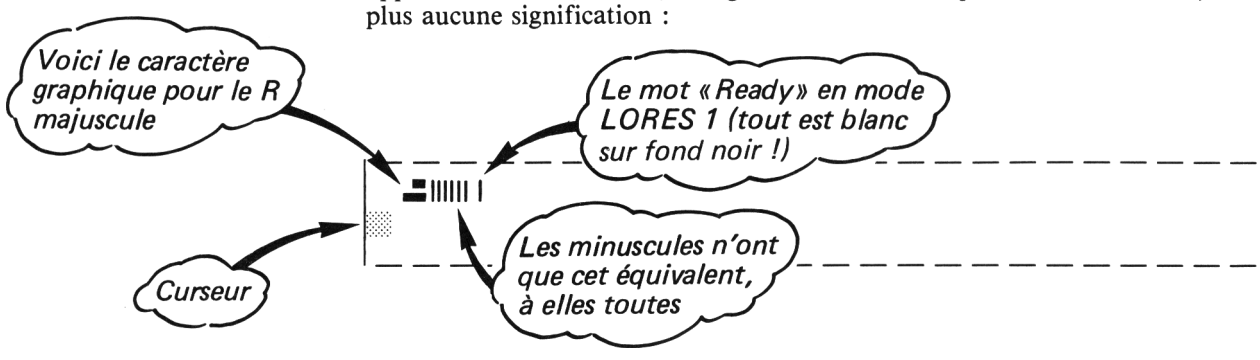
5. Le mode LORES 1

LORES 1

Passons au mode LORES 1. Dans ce mode, la différence avec LORES 0 réside dans le fait qu'*on change de jeu de caractères.* Vous ne disposez plus des caractères alphanumériques mais de caractères spéciaux, appelés des *caractères graphiques :* un demi-pavé, ou un quart, etc. A chaque touche correspond l'un de ces caractères. Faites, en mode direct et après un CTRL/L :

```
LORES 1
```

Puis, frappez RETURN. L'écran devient noir et un mot qui devrait être « *Ready* » apparaît en blanc ; en fait, il s'agit des dessins correspondant à ces lettres ; ils n'ont plus aucune signification :



CARACTERES GRAPHIQUES

Ces caractères graphiques ont pour numéro de caractère 32 à 128 en *version dite alternée.* Vous pouvez les afficher, mais en utilisant un code « secret » qui est ,CHR$(27);"I" et que comprend votre Basic. Par exemple :

```
? CHR$(65)
A
```

Le A est le code ASCII n°65

Mais, par contre et avec le code d'alternative :

```
? ,CHR$(27);"I";CHR$(65)
```

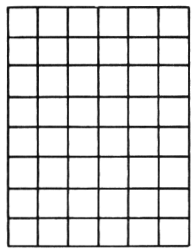
C'est l'autre code, «alterné», graphique du n° 65

Pour afficher sur un seul écran tous les caractères graphiques, de 32 à 128, créez ce programme et lancez-le (après un NEW, éventuellement) :

```
10 FOR N=32 TO 128
20 ? ,CHR$(27);"I";CHR$(N)
30 NEXT N
```

Code pour les caractères alternatifs

Tous ces caractères s'inscrivent individuellement dans une « grille » de 8 par 6 qu'on peut représenter ainsi :



Si tout est plein, vous obtenez le pavé correspondant au curseur ; selon le remplissage, on forme tous les autres caractères, dont les graphiques.
Pour faire défiler tous ces caractères graphiques *avec* leur numéro, entrez ce programme :

```
10 FOR N=32 TO 128
20 ? N,CHR$(27);"I";CHR$(N)
30 ?
40 WAIT 35
50 NEXT N
```

Pour sauter une ligne ...

On a simplement ajouté N pour afficher le code

... Et temporiser un instant

Il y a 96 caractères et l'on en place un toutes les deux lignes, de sucroît : vous ne les verrez pas tous d'un coup.

CTRL/C

CONT

Pour stopper leur défilement, frappez sur CTRL/C. Pour le faire reprendre, faites CONT pour CONTinuer, en frappant ces lettres sur le clavier. Cette méthode s'applique très généralement à tous les cas de défilement.

Au passage, remarquez le caractère 57, qui ressemblerait à une fusée : voilà qui peut être utile pour les jeux ! Celui qui est semblable au curseur est le 95. Le 83 est un gros signe « égal » (soit =). Utilisons-le pour le renvoyer d'un bord à l'autre de l'écran, en rebondissant.

Il faut, tout d'abord, le placer sur la rangée choisie, 12 par exemple, en position de départ, à gauche, sur la colonne C = 1, par exemple. Puis, on va l'emmener de C = 1 à C = 2 *mais il faudra l'effacer* sur C = 1. Pour l'effacer, on affichera simplement un espace, donc " " sur la position antérieure. Démarrons ce programme en commandant un mouvement de gauche à droite (faites un NEW au préalable) :

```
10 REM mouvement
20 CLS
30 LORES 1
40 FOR C=1 TO 38
50 PLOT C,12,CHR$(83)
60 PLOT C-1,12," "
70 NEXT C
```

Pour placer le caractère 83, qui est un gros "="

Pour l'effacer de la position précédente (C–1) et donner l'illusion du mouvement

MOUVEMENT

Lancez ce programme ; le « 83 », un gros signe égal, va se déplacer de la gauche à la droite de l'écran où il s'arrêtera. Il faut maintenant le faire revenir ; pour cela, le FOR... TO va aller *de* 37 *à* 0 par pas négatif de – 1. L'effacement, lui, ne se fera non plus sur C – 1 mais sur C + 1 puisqu'on fait marche arrière. Cela donne :

On recule, cette fois

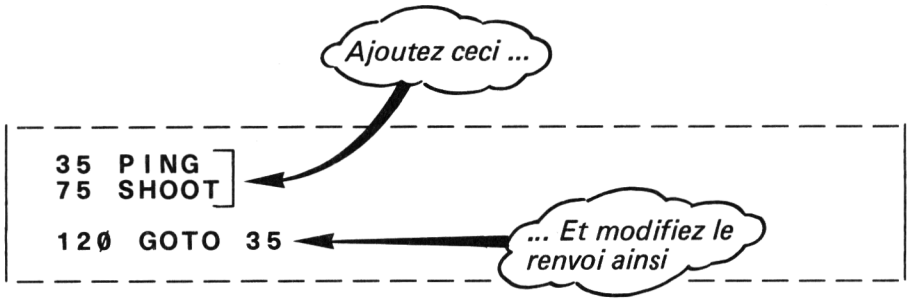
```
80 FOR C=37 TO 0 STEP -1
90 PLOT C,12,CHR$(83)
100 PLOT C+1,12," "
110 NEXT C
120 GOTO 40
```

Nouvel affichage ...

... Et effacement de la position précédente

Pour recommencer de l'avant!

Lancez ce programme ; incontestablement, l'illusion du mouvement, de la gauche vers la droite et retour, et on repart, etc., est correcte. Alors, ajoutons le son, une sonnette au départ de la gauche, un SHOOT à l'arrivée à droite :



Le programme complet est ainsi :

```
10 REM mouvement
20 CLS
30 LORES 12
35 PING
40 FOR C=1 TO 38
50 PLOT C,12,CHR$(83)
60 PLOT C-1,12," "
70 NEXT C
75 SHOOT
80 FOR C=37 TO 0 STEP -1
90 PLOT C,12,CHR$(83)
100 PLOT C+1,12," "
110 NEXT C
120 GOTO 35
```

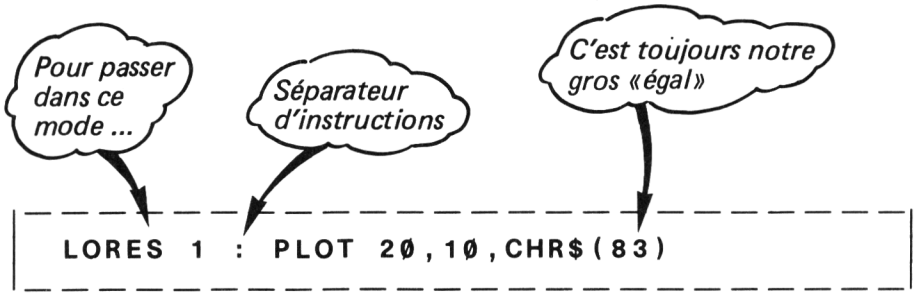
Lancez-le : c'est plus réaliste avec le son. Maintenant, à vous de tenter de l'améliorer, avec de la couleur, par exemple. Essayez : pourquoi pas ?

6. Pour mêler les genres

Sachez qu'on peut encore employer les caractères alphanumériques en mode LORES 1 ou ceux graphiques en mode LORES 0, et ce, grâce à un caractère spécial qui est, selon le cas :

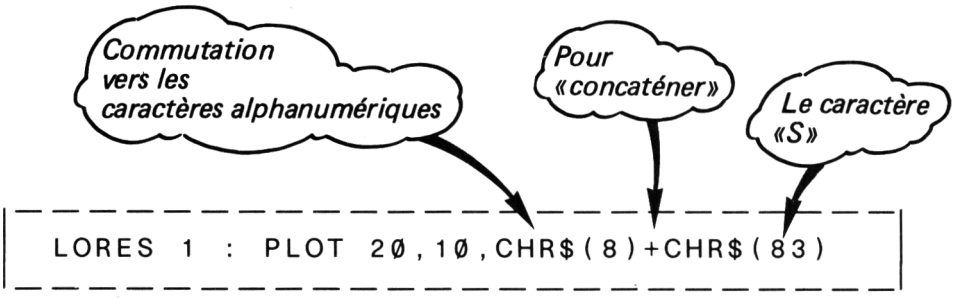
• Pour obtenir les caractères *alphanumériques* en LORES 1, il faut donner le code CHR$(8). Faites l'essai, d'abord en LORES 1 « pur » et en mode direct.

CHR$(8)
CHR$(9)



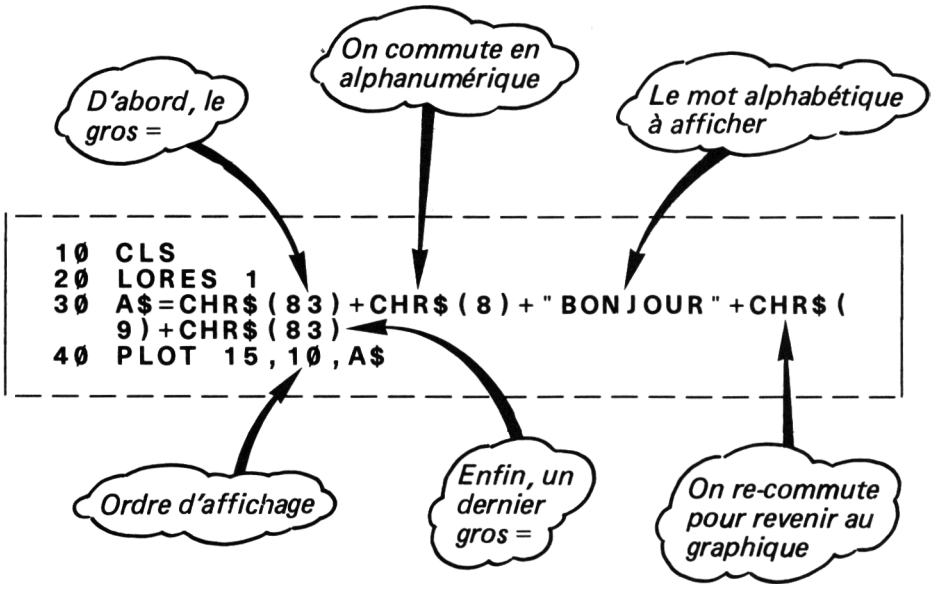
Au RETURN, le même symbole graphique va s'afficher, en blanc sur fond noir, à la position 20, 10.

Bien : maintenant, on va rester en LORES 1 pour afficher le caractère *alphanumérique* correspondant au même code 83 ; c'est un S majuscule. Pour cela, on va « commuter » en alphanumérique avec CHR$(8) puis, à l'aide du caractère de *concaténation,* le symbole « plus » (+) qu'on vous a présenté au début de ce livre, on va demander l'affichage du S, ce qui donne (faites d'abord un CTRL/L) :



Au RETURN, c'est un bel S qui s'affiche, cette fois, en 2Ø,1Ø.

• Pour revenir aux caractères graphiques, on « commuterait » à nouveau « le clavier » avec, cette fois, +CHR$(9). Voici un petit programme qui vous en indique le mode d'emploi :

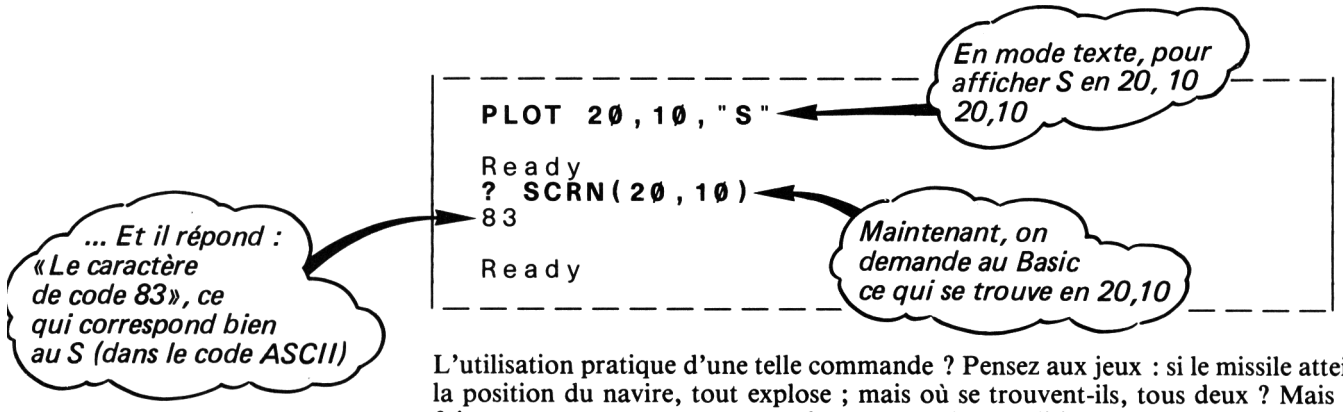


On a utilisé la variable alphanumérique A$, définie en 3Ø, pour bien vous faire comprendre le processus (car ainsi, on peut l'isoler de l'ordre PLOT). Au RETURN, vous verrez s'afficher à partir de 15,1Ø en blanc sur fond noir :

• En mode LORES Ø, pour obtenir les caractères graphiques, on procède de même mais en inversant les commandes. Ainsi, CHR$(9) fournit les caractères graphiques alors que CHR$(8) ramène au « clavier » alphanumérique.

Considérez CHR$(8) et CHR$(9) comme des commutateurs :
• CHR$(8) commute vers le clavier alphanumérique.
• CHR$(9) commute vers le clavier graphique.

Expérimentons un dernier ordre qui s'applique aussi bien en mode TEXT qu'en LORES Ø et LORES 1. Passons alors simplement en TEXT, avec CTRL/L, et plaçons un S en 2Ø,1Ø avec un PLOT. Puis, on va demander au Basic quel caractère se trouve en 2Ø,1Ø (en supposant qu'on l'ait oublié). Cette demande se dit PRINT SCRN(2Ø,1Ø), où SCRN vient de « *screen* », écran en anglais. Voici ce que cela donne :



L'utilisation pratique d'une telle commande ? Pensez aux jeux : si le missile atteint la position du navire, tout explose ; mais où se trouvent-ils, tous deux ? Mais au fait, comment programme-t-on des commandes conditionnelles de ce type ? Vous l'apprendrez dans notre prochain chapitre.

QUESTIONS SUR LE CHAPITRE VI

Voici quelques questions, mais en fait, nous souhaitons que vous vous entraîniez ensuite à imaginer des programmes par vous-même !

1. En mode graphique LORES, la résolution de l'écran (le nombre de positions) est de :
   ☐ 39 fois 27
   ☐ 620 par 340

2. Peut-on écrire : PLOT "BONJOUR"
   ☐ Oui
   ☐ Non

3. Pour placer une couleur en LORES et en un point précis, l'ordre est :
   ☐ PLOT colonne,rangée,couleur
   ☐ PLOT couleur = rangée;colonne

4. La différence entre LORES Ø et LORES 1 est constituée par :
   ☐ La définition de la grille disponible
   ☐ Le jeu de caractères

5. Peut-on indiquer le numéro de ligne avec un RUN pour choisir entre plusieurs programmes ?
   ☐ Oui
   ☐ Non

6. A quoi sert l'ordre END :
   ☐ A marquer la fin d'un programme pour le Basic
   ☐ A séparer des séquences de programmes lors de leur rédaction, afin que le programmeur s'y retrouve.

7. Pour appeler un sous-programme, on doit écrire :
   ☐ GOTO SUB *numéro*
   ☐ GOSUB *numéro*
   où *numéro* est le numéro de l'instruction à laquelle commence le sous-programme

8. A la fin du sous-programme, l'instruction de retour s'écrit :
   ☐ RETURN *numéro de ligne*
   ☐ RETURN

9. Pour arrêter un défilement sur l'écran, on frappera :
   ☐ RESET
   ☐ CTRL/C

Réponses page 120

REPONSES AUX QUESTIONS SUR LE CHAPITRE VI

1. En mode LORES, la grille d'écran est de 39 colonnes sur 27 rangées.

2. Non, la syntaxe est fausse car il faut indiquer la position.

3. L'ordre correct est PLOT colonne,rangée,couleur.

4. Le jeu de caractères n'est pas le même en LORES Ø et en LORES 1.

5. Oui ; on écrira RUN 5ØØ, par exemple, pour un programme commençant en 5ØØ si un autre le précède.

6. END marque la fin d'un programme pour le Basic. Mais vous pourrez aussi l'employer pour tester des séquences de programme que vous venez de rédiger : placez *provisoirement* un END là où vous voulez qu'elles s'arrêtent ; supprimez-le ensuite.

7. Pour appeler un sous-programme, on écrit GOSUB *numéro,* par exemple GOSUB 199Ø.

8. A la fin d'un sous-programme, l'instruction de retour s'écrit RETURN tout court, surtout sans numéro de ligne !

9. Pour arrêter un défilement sur l'écran, faites CTRL/C.

13

ARC-EN-CIEL

LORES PLOT READ DATA

L'instruction PLOT permet de créer très vite une bande horizontale dont on précise la couleur. Nous allons l'exploiter pour obtenir une palette de couleurs sous forme de bandes s'étalant sur plusieurs lignes de l'écran. Pour commencer, suivons l'ordre des couleurs convenu par l'Oric ;
1 rouge, 2 vert, 3 jaune, 4 bleu foncé, 5 magenta, 6 bleu clair.

Dans l'instruction PLOT, il faut ajouter 16 pour obtenir le code de couleur du papier.

```
10 CLS
20 L=0
30 FOR C=1 TO 6
40 FOR N=1 TO 4
50 PLOT 0,L,C+16
60 L=L+1
70 NEXT N
80 NEXT C
```

REMARQUES

20 — on commence à prendre dès le haut de l'écran

30 à 80 — boucle sur les 6 couleurs

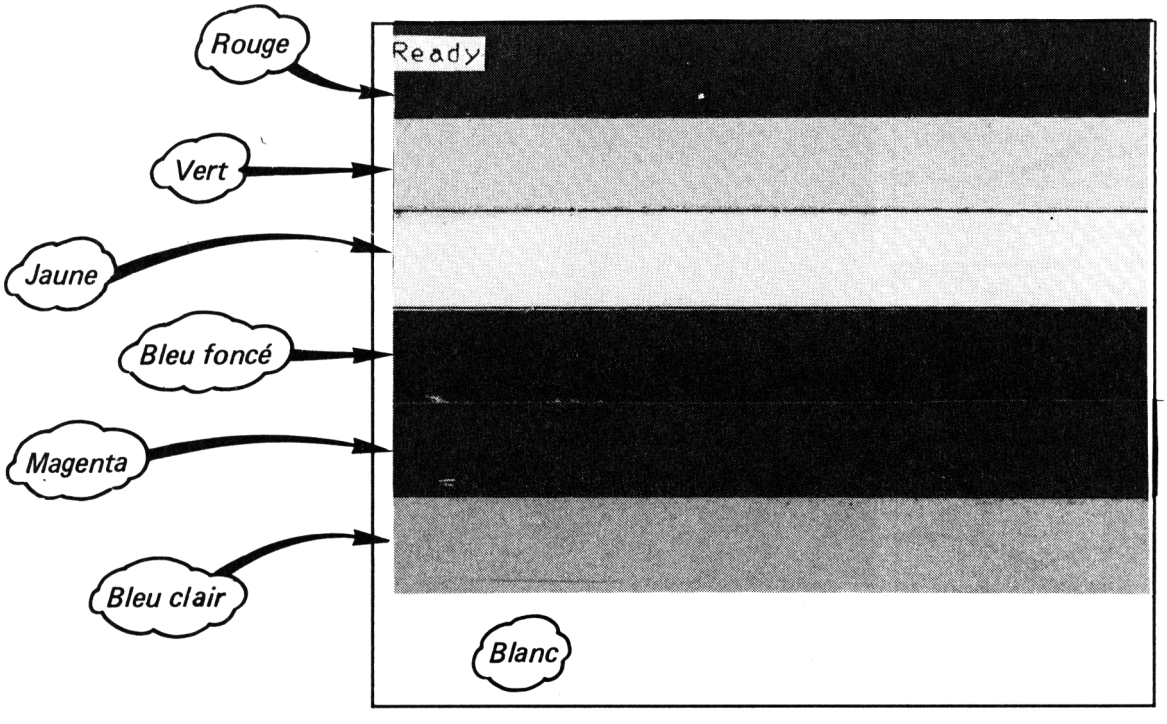
40 à 70 — boucle pour une bande (l'équivalent de 4 lignes de texte)

50 — apparition d'une ligne colorée

60 — passage à la ligne suivante

L'exécution de ce programme va vous donner 6 bandes de couleur et une bande blanche en bas de l'écran.

Le message Ready s'affiche vers le haut de l'écran puisqu'on n'a rien écrit depuis l'instruction d'effaçage de la ligne 10. PLOT ne déplace pas le curseur comme le fait PRINT.



Si l'on veut ranger les couleurs, comme elles figurent dans l'arc-en-ciel : violet, indigo, bleu, vert, jaune et rouge, il faut utiliser les codes dans cet ordre. Pour faire respecter cette succession, nous utiliserons deux instructions nouvelles qui vont de pair : READ et DATA.

READ
DATA

DATA permet de constituer très facilement un fichier de valeurs, et READ lit ces valeurs. Si on écrit DATA 5,4,3,2,1 puis READ CO c'est la valeur 5 qui est lue et placée dans CO, comme si on avait écrit CO = 5.

Si une deuxième lecture intervient avec READ CO, c'est la valeur 4 qui se met dans CO en effaçant la valeur 5 qui s'y trouvait comme si on avait écrit C0 = 4.

Mais la grande différence, c'est que si l'instruction READ CO est placée dans une boucle, elle lira successivement les valeurs figurant dans l'instruction DATA.

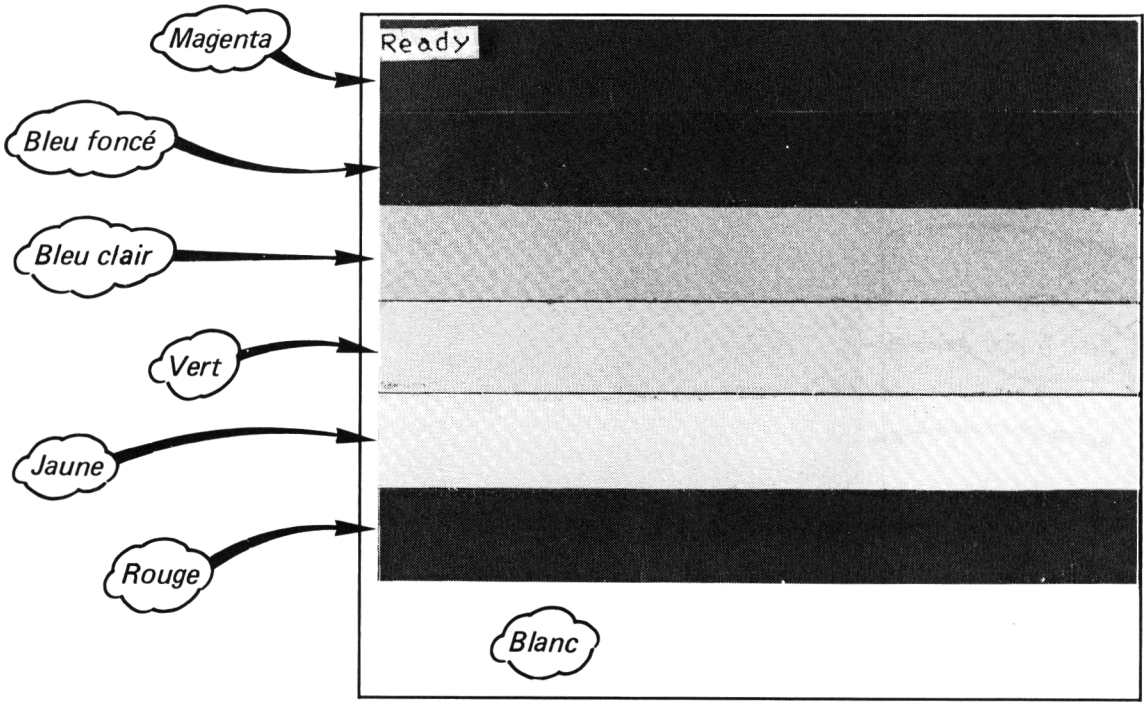
READ CO est général puisque la valeur ne figure pas, contrairement à CO = 4 ou CO = 5. On retrouve la même souplesse que pour l'instruction INPUT que l'auteur du programme emploie pour laisser la parole à l'utilisateur, mais avec READ c'est l'auteur qui fixe les données.

```
10 CLS
15 DATA 5,4,6,2,3,1
20 L=0
30 FOR C=1 TO 6
35 READ CO
40 FOR N=1 TO 4
50 PLOT 0,L,CO+16
60 L=L+1
70 NEXT N
80 NEXT C
```

REMARQUES

15 — les couleurs dans l'ordre

35 — lecture d'un code d'une couleur



LORES

Jusqu'à maintenant on a travaillé en mode texte. Que se passe-t-il si dans ce programme, on remplace CLS par LORES 0, à la ligne 10 ?

L'instruction PLOT n'agit plus sur toute la partie de la ligne située à droite de la case précisée, mais seulement sur cette case.

Le programme précédent va tracer une colonne colorée comme les bandes, sur fond noir, et le mot Ready s'affichera en bas de l'écran (en chassant d'ailleurs la moitié de la colonne violette).

Si on change la ligne 40 :

```
40 PLOT 20,L,CO+16
```

la colonne se déplace au milieu de l'écran.

Le mode LORES 1 donne le même résultat, puisqu'on ne définit que la couleur du papier, sans tracer de caractères.

Vous avez remarqué que le retour en mode Texte se fait automatiquement lorsqu'on atteint le bas de l'écran.

Un drapeau

Pour beaucoup de drapeaux, les bandes sont verticales. Notre programme ne convient donc pas et nous avons le choix entre deux méthodes : dessiner les bandes en mode graphique basse résolution en juxtaposant plusieurs colonnes de même couleur pour obtenir une bande large, ou bien rester en mode texte : pour un drapeau tricolore, on remplira d'abord l'écran entièrement d'une couleur, puis en se décalant vers la droite, on modifiera la couleur des deux tiers de droite, pour finir avec le dernier tiers.

Prenons le cas du drapeau français avec des bandes larges de 13 cases :

```
10 CLS
20 PAPER 0:INK 7
30 DATA 6,7,1
40 FOR C=1 TO 3
50 READ CO
60 FOR L=0 TO 25
70 PLOT (C-1)*13,L,CO+16
80 NEXT L
90 NEXT C
```

REMARQUES

20 — pour avoir un fond noir

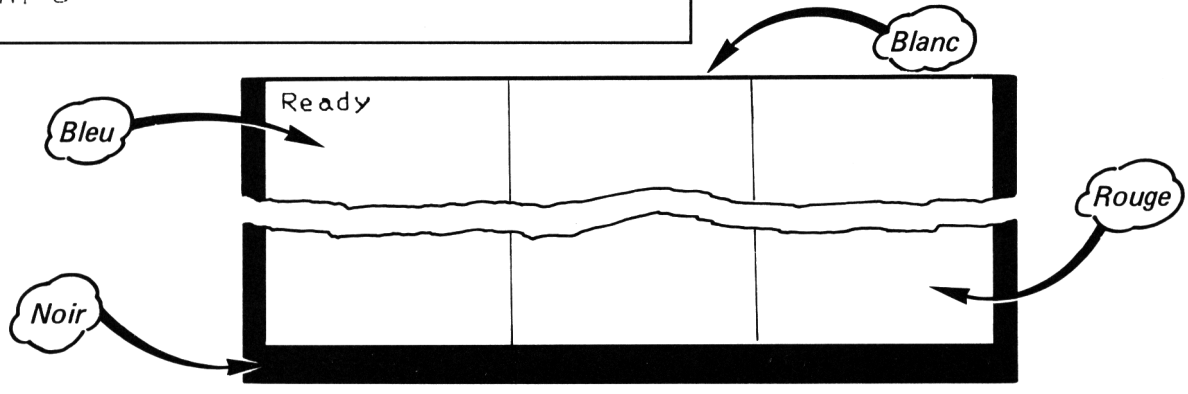
30 — les couleurs, bleu, blanc, rouge

40 à 90 — boucle sur ces trois couleurs

50 — lecture d'une couleur

60 à 80 — boucle pour une bande : 26 lignes de haut

70 — coloration d'une ligne, ou des deux tiers de droite, ou du tiers



Vous pourrez passer en revue tous les drapeaux tricolores à bandes verticales, en changeant simplement les valeurs de la ligne 30.

14

C'EST PLUS C'EST MOINS
PLOT SGN

Ce jeu très connu peut bénéficier d'une présentation visuelle en se servant de l'instruction PLOT et en regardant l'écart entre le nombre à découvrir et celui que vous proposez.

Si cet écart est plus petit que zéro, vous êtes trop bas et vous obtiendrez sur l'écran une ligne rouge, si vous êtes trop haut, la ligne sera jaune, et si vous avez trouvé elle sera verte. Vous avez droit à 20 essais, même si vous trouvez avant.

Pour arrêter avant le vingtième essai, tapez [CTRL] [C], en maintenant la touche [CTRL] enfoncée au moment où vous appuyez sur [C].

SGN ()

On utilisera une fonction nouvelle, SGN, pour savoir si l'écart est plus petit ou plus grand que zéro, ou s'il est nul : SGN(−2) vaut −1, SGN(3) vaut 1 et SGN(0) vaut 0.

```
10 NM=25
20 CLS:PRINT"DEVINEZ UN NOMBRE DE 1 A
";NM
30 A=INT(RND(1)*NM)+1
40 FOR E=1 TO 20
50 INPUT N
60 Q=SGN(A-N)
70 P=(NM-N)*26/(NM-1)
80 PLOT 10,P,18-Q
90 PLOT 11,P,STR$(N)
100 NEXT E
```

REMARQUES

10 — le plus grand nombre possible

30 — nombre à découvrir compris entre 1 et NM, inclus

40 à 100 — boucle donnant droit à 20 essais

50 — introduction d'une proposition

60 — Q = 1 : c'est moins, Q = −1 : c'est plus, Q = 0 c'est trouvé

70 — P : numéro de la ligne à colorier, entre 0 et 26

80 — apparition de la bande teinte en rouge, jaune ou vert selon le cas

90 — affichage du nombre proposé au début de la bande

Utilisation :

La ligne 20 fixe une limite pour le nombre à découvrir qui sera toujours compris entre 1 et cette limite. Vous pouvez la changer mais il faut que les nombres proposés la respecte, sinon vous obtiendrez une erreur : Illegal quantity in 80.

Pour que le programme s'arrête de lui-même lorsque le nombre est trouvé, il faut ajouter une ligne dans la boucle qui force l'arrêt si Q vaut zéro. Ce serait le rôle de l'instruction IF... THEN... que vous rencontrerez dès le chapitre VII. Si vous savez déjà l'utiliser, n'hésitez pas à vous en servir dès maintenant !

15

UNE MOSAÏQUE VIVANTE
PLOT LORES

En mode texte l'instruction PLOT peut servir à fixer la couleur du papier sur la partie droite de la ligne qui commence à la case précisée. Si on l'emploie un grand nombre de fois en changeant la case et la couleur, on obtiendra une espèce de mosaïque formée de bandes horizontales de largeur et de couleurs variées. Laissons le hasard déterminer les changements : la couleur, la place horizontale et verticale de la case de gauche.

```
10 CLS
20 CO=INT(RND(1)*8)
30 C=INT(RND(1)*39)
40 L=INT(RND(1)*27)
50 PLOT C,L,CO+16
60 GOTO 20
```

REMARQUES

20 — la couleur

30 — numéro de colonne

40 — numéro de ligne

50 — une bande

60 — on continue

Pour représenter les couleurs voici ce qu'on a trouvé : une bande de couleur c'est un chiffre suivi de @

```
@04@@1@@7@70@0@6@3@6@@6@@1@@62@@11@@@1
54@2@@@@44@@@75@@@4@7@@@65@@@57@6@@@5@
@@5@@23@@@@@@@2@04@@4@612@@@@@@7@0@@6@
@2@@@@0@@@@@0@5@@@@@0@4@@@2@3@@@@5@@@@
@3@@3@@@@@3@@5@5@@7@@@1@@@0@@@@@@0@6@20
@4@2@4@61@1@@72@6@@@@5@@@3@@5@@715@@@3
5@@@3@77@@7@@@@464@@5@@@@7@@1@71@7@@20
@@@@@@@@@@54@@346@0@@@@@@@@4@34@@@@7@7
@@@@5@@30@@4433@@5@1741@6@1@@@@6@@@@@1
5@@@22@50@0@1@@@@@@@@6@5@@60@1@06@@55@
@4@@@56@0@@@@1@@54@@@@73@@@2@456@2@@2@
@0@@@2@0@0@@@@@@@73@32@@@@0@@6@77@75@@
@@@21@@@5@07@@14@@@@@6@4425@@@@@15@@@7
22@@@@@0@@@00@@@@@@@44@@3@2@@@@5@@@23@
@@6@73@@0@@554@2415@@@@@@@31@@@61@@@5@
05@36@3@7@7@@@@74@@5@@@6470@@3@@@@01@@
@500@@@10@36@@@6@@1621@7@0@3@2@@@@@433
@@1@@@@1@@@@@@@@2@@71@@@0@@301@337@@@@
@@33@@@5@@@@@@2@@@@2@@21@7@@@@2@@62@30
@@@26@@@1@@@1@@12@47@7@@@@@@@@@1@@@412
@507@13@6@@@@7@@@1@7@@@@06@@2@@55@6@@@
@1@4@1@351@@30@4@2@@@@@@@3@0@@@4@@@@27
@@@006@@@@@@@@@@6@@@@4501@@155@@2@@@@@
@@3@4176@1@45@4@@710@@2@@@@@1@201@5@@6
411@@@3@3@@@@@@@@@@64@71@0@@7@@3@3@@40
5@@@65@51@@@@3@@1@@@@10@@5@51@@626@31@
@62@1@@@@@@@@@@@37@@71@@@@@@5370627@@@
```

1 @@@@ c'est une bande rouge qui occupe la place de 5 caractères

Un carré bleu foncé

Chaque nouvelle bande détruit ce qui se trouvait à sa place, jusqu'au bord droit de l'écran. Pour composer la mosaïque avec des carreaux et non des bandes, il suffit de se placer en mode graphique faible résolution. L'instruction PLOT ne concerne plus alors que la case dont on fournit la position.

```
10 LORES 0
```

REMARQUE

10 — mode graphique basse résolution

Le fond est noir et se recouvre progressivement de carreaux de couleur, dont le noir puisque la ligne 20 donne à CO des valeurs allant de 0 à 7.

```
20407040310006306060700002002146003166
17152012210042060170052032426010003001
73040051027000400420060006300607726710
00727670223005044347601600053006304640
10020401003136210105005200202763560652
76500002406241063104130505146053300172
52206003305450020063710013006303706477
14044255530705440305514000230704040625
13365700510603043035471761565050041462
62132514031406000105200120440262034060
06003150515020000754415075073700620100
00003005220411707007760715400503173706
40002450252070706034006001206477506056
04204360120014206010030555551600253325
20007031061425607010350130206570030010
60003537200300303004760177015402061051
10000400401301500340005174600301607300
11030302000700131344412144300600406272
14700552266157203010251060515462120360
01706677300701045602060603000001035400
20024600046020142063254130661436072365
41667520400615250063750333005003205310
70360605360607004010001030065066004755
54201100666700021560001104001020120206
47647102750402307160407004036602670701
00010073400731700106560646533617665205
40006505040730100310200563534270020067
```

Tout au long du spectacle, un carré blanc ne cesse de clignoter : c'est le curseur. Pour le supprimer on peut ajouter la ligne :

```
5 PRINT CHR$(17)
```

Quand vous aurez arrêté le programme avec CTRL C , tapez CTRL Q pour récupérer le curseur.

16

HISTOGRAMME
LORES PLOT DIM

Un histogramme est un diagramme à barres c'est-à-dire un moyen de représenter graphiquement l'évolution d'un phénomène au cours du temps, une répartition en différentes catégories...

Ce diagramme a l'aspect de tuyaux d'orgue : des barres verticales dont les hauteurs variées traduisent l'importance du phénomène à représenter.

L'instruction PLOT utilisée en mode graphique basse résolution permet ce genre d'image. Si on accepte de compter les valeurs et de chercher la plus grande d'entre elles, le programme qui va tracer l'histogramme peut se découper en deux parties : l'introduction des données, et le tracé. Nous traiterons un maximum de 30 valeurs et la barre la plus haute occupera 27 lignes.

Avec une seule couleur :

```
10 CLS
20 INPUT"NOMBRE DE VALEURS";NB
30 DIM V(NB)
40 INPUT"VALEUR MAXIMUM";VM
50 FOR N=1 TO NB
60 PRINT"VALEUR NO ";N,
70 INPUT V(N)
80 NEXT N
90 CLS
95 :
100 LORES 0
110 FOR N=1 TO NB
120 NL=INT((VM-V(N))*26/VM)
130 FOR L=NL TO 26
140 PLOT N+8,L,17
150 NEXT L
160 NEXT N
```

REMARQUES

30 — réservation de places en mémoire vive pour les valeurs

40 — VM servira à cadrer l'histogramme : la valeur la plus grande en haut de l'écran, zéro en bas

50 à 80 — boucle d'introduction des NB valeurs

90 — effacement de l'écran en mode texte pour positionner le curseur en haut à gauche

100 — passage au mode graphique basse résolution

110 à 160 — boucle sur les valeurs

120 — position du haut d'une barre

130 à 150 — boucle de tracé d'une barre

140 — barre rouge

DIM

A la ligne 30 apparaît une instruction nouvelle : DIM. Elle permet de prévoir de la place en mémoire vive pour un fichier dont le nom sera V et qui contiendra les valeurs introduites à la ligne 70.

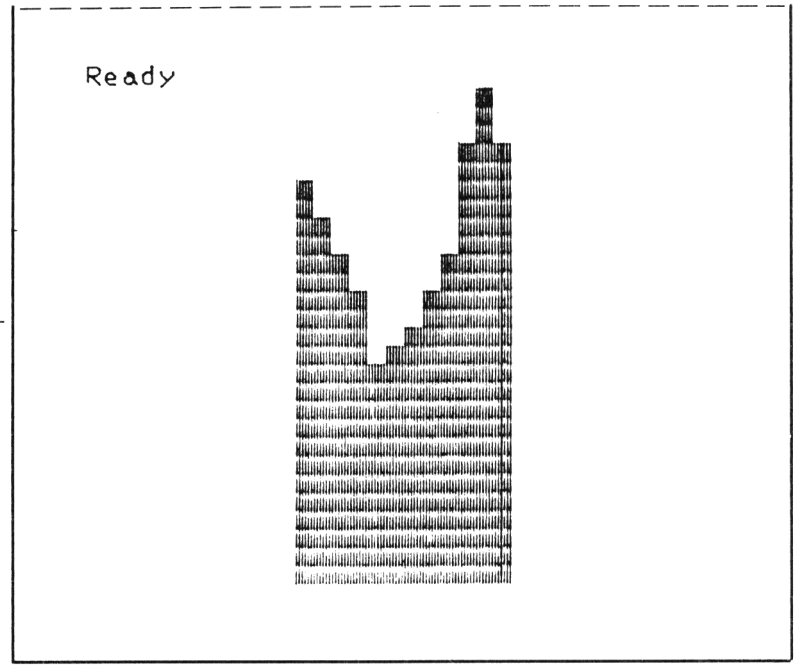
Ces différentes valeurs portent le même nom, V, et on les distinguera par un numéro placé entre pharenthèses (lignes 70 et 120)

Exemple :

Représenter sous forme d'histogramme la hauteur d'eau tombée en moyenne chaque mois à Brest.

Janvier	80 mm	Juillet	50 mm
Février	72 mm	Août	55 mm
Mars	63 mm	Sept.	65 mm
Avril	54 mm	Octobre	85 mm
Mai	40 mm	Novembre	100 mm
Juin	45 mm	Décembre	85 mm

```
NOMBRE DE VALEURS? 12
VALEUR MAXIMUM? 100
VALEUR NO 1     ? 80
VALEUR NO 2     ? 72
VALEUR NO 3     ? 63
VALEUR NO 4     ? 54
VALEUR NO 5     ? 40
VALEUR NO 6     ? 45
VALEUR NO 7     ? 50
VALEUR NO 8     ? 55
VALEUR NO 9     ? 65
VALEUR NO 10     ? 85
VALEUR NO 11     ? 100
VALEUR NO 12     ? 85
```



Comme toujours lorsque le programme s'achève, le mot Ready apparaît, en-dessous de la dernière ligne affichée par une instruction PRINT. C'est grâce à la ligne 90 que Ready s'affiche en haut de l'écran, sinon il pourrait très bien apparaître en bas et même faire glisser l'histogramme vers le haut ! Pour éviter que Ready ne coupe une ou plusieurs barres, on a décalé le début de l'histogramme en ajoutant 8 à N à la ligne 140.

Version multicolore

Dans cette ligne la couleur du papier est toujours le rouge (code 17). Pour faire ressortir encore mieux les diverses amplitudes du phénomène observé, on peut changer de couleur de temps en temps. On a choisi de le faire toutes les 4 lignes et de tolérer toutes les couleurs sauf le noir qui est déjà la couleur du fond ! La couleur d'une colonne dépendra de sa hauteur.

Voici les changements à apporter :

```
120 NL=INT((VM-V(N))*26/VM)
125 T=INT(NL/4)
130 FOR L=NL TO 26
140 PLOT N+8,L,17+T
150 NEXT L
```

REMARQUES

125 — T est la teinte d'une barre

140 — barre de couleur variée

Nous ne vous montrerons pas de résultats, couleur oblige !

17

UN MOT TRANSFORMÉ
PLOT SCRN

SCRN

Nous avons surtout utilisé les lettres majuscules, et pourtant votre Oric sait très bien afficher les minuscules, car elles ont leur code tout comme leur grande sœur. Le code du *a* est 97 celui du *A* 65, pour *b* c'est 98 et pour *B* 66, c'est-à-dire qu'il suffit d'ajouter 32 au code de la majuscule pour obtenir celui de la minuscule. On va même observer la transformation sur l'écran : la fonction SCRN permet de connaître le code d'un caractère affiché sur l'écran. Affichons d'abord un mot en majuscules, puis grâce à SCRN, modifions le code de chaque caractère affiché.

```
10 CLS
20 INPUT"VOTRE MOT";M$
30 PLOT 5,6,M$
40 WAIT 50
50 FOR C=1 TO LEN(M$)
60 CA=SCRN(C+4,6)
70 PLOT C+4,6,CHR$(CA+32)
80 NEXT C
```

REMARQUES

30 — affichage du mot sur la 6e ligne à partir de la cinquième colonne

40 — une pause pour voir

50 à 80 — boucle pour changer chaque lettre

60 — CA : le code de la Ce lettre du mot

70 — affichage de la lettre transformée

Tapez un mot, rien d'autre que des lettres.

Votre mot s'affiche sur la sixième ligne, reste un instant puis se transforme lettre après lettre :

```
VOTRE MOT? ALPHABETISATION

Ready

alphabetisation
```

Après transformation

Si vous tapez des caractères qui ne sont pas des lettres, le changement risque d'être étonnant : une virgule se transforme en L !

Comment conserver l'initiale en majuscule ? Il suffit de commencer la transformation à partir de la deuxième lettre :

```
50 FOR C=2 TO LEN(M$)
```

```
VOTRE MOT? ALPHABETISATION

Ready

Alphabetisation
```

18

FAIRE-PART DE NAISSANCE

PLOT GOSUB RETURN

Comment écrire un mot ou une phrase en caractères de taille double ? En occupant deux lignes au lieu d'une aurait dit Monsieur de la Palice ! C'est en effet ce qu'il faut faire en prévenant sur chaque ligne que ce qui suit doit être soit le haut soit le bas du texte. Voici un programme très court qui affichera en gros caractères le texte que vous introduirez. Nous verrons après une application plus complète.

```
10 CLS
20 INPUT"TEXTE A TRAITER";T$
30 PLOT 10,3,CHR$(10)+T$
40 PLOT 10,4,CHR$(10)+T$
```

On écrit deux fois le texte en mémoire. Vous obtiendrez ...

REMARQUES

20 — votre texte

30 — affichage de la partie supérieure

40 — idem pour le bas du texte

... sur l'imprimante deux fois la même ligne et sur l'écran une seule ligne mais en double hauteur

Le texte apparaît au début des lignes 5 et 6 (celle du haut comptant pour zéro).

```
TEXTE A TRAITER? IL ETAIT UNE FOIS..

Ready
          IL ETAIT UNE FOIS..
          IL ETAIT UNE FOIS..
```

IL ETAIT UNE FOIS..

Dans les lignes 30 et 40 c'est le code 10 qui est fourni à la fonction CHR$ pour obtenir la double hauteur. Si on place 14 comme code, on obtient en plus un affichage clignotant, que nous ne vous montrerons pas...

Le faire-part de naissance

Ce procédé de grossissement est commode pour mettre en relief certaines parties du texte. Nous allons l'employer pour faire ressortir le nom et le prénom du nouveau-né dans le faire-part qui annonce sa naissance.

Tapez le nom de famille, le prénom et la date et vous obtiendrez une carte facile à lire ! C'est encore une fois l'instruction PLOT qui va nous servir aussi bien pour écrire normalement qu'en double hauteur. Et puisque le même travail revient plusieurs fois dans chacun des deux cas, nous fabriquons deux outils, deux sous-programmes, un pour chaque mode d'écriture.

Pour la taille normale il suffit d'afficher sur une ligne et de dire qu'il faut poursuivre à la ligne suivante.

Pour la double hauteur, reprenons le programme de tout à l'heure en ajoutant deux précautions. Si la ligne du haut porte un numéro pair tout va bien, sinon le haut et le bas sont inversés : amusant ! Et puis il faut dire aussi de poursuivre deux lignes plus loin.

Nous ferons appel à l'un de ces deux sous-programmes chaque fois qu'il faudra afficher une ligne.

```
10 CLS
20 PRINT"JE VAIS VOUS FAIRE UN JOLI FA
IRE-PART"
30 PRINT"DE NAISSANCE SI VOUS CONSENTE
Z A "
40 PRINT"REPONDRE A TROIS PETITES QUES
TIONS .."
50 INPUT"NOM DES HEUREUX PARENTS ";N$
60 INPUT"PRENOM DU BEBE ";P$
70 INPUT"ET IL EST NE QUAND CE PETIT "
;D$
80 CLS
90 L=3
100 T$="M. et Mme "+N$
110 GOSUB 240
120 T$="ont la joie de vous annoncer"
130 GOSUB 210
140 T$="la naissance de leur bebe"
150 GOSUB 210
160 T$=P$
170 GOSUB 240
180 T$="le "+D$
190 GOSUB 210
200 END
205 REM----------------------
210 PLOT 2,L,T$
220 L=L+1
230 RETURN
235 REM----------------------
240 L=2*(INT(L/2))+1
250 PLOT 2,L,CHR$(10)+T$
260 PLOT 2,L+1,CHR$(10)+T$
270 L=L+2
280 RETURN
```

REMARQUES

20 à 40 — le mode d'emploi

30 à 70 — introduction des trois informations

90 — L fixe le début du faire-part : ligne 3 (à partir du zéro)

100 — texte de la première ligne

110 — utilisation du sous-programme double hauteur

120 et 130 — 2e ligne : simple hauteur

140 et 150 — idem pour la 3e ligne

160 et 170 — le prénom en gros

180 et 190 — la date en plus petit

200 — fin du programme, les sous-programmes vont suivre

210 à 230 — sous-programme simple hauteur

210 — affichage d'une ligne

220 — position de la ligne suivante

240 à 280 — sous-programme double hauteur

240 — position du haut du texte : nombre impair

250 et 260 — le texte sur deux lignes

270 — position de la ligne suivante

```
JE VAIS VOUS FAIRE UN JOLI FAIRE-PART
DE NAISSANCE SI VOUS CONSENTEZ A
REPONDRE A TROIS PETITES QUESTIONS
NOM DES HEUREUX PARENTS ? RAVIS
PRENOM DU BEBE ? DESIRE
ET IL EST NE QUAND CE PETIT ? 31 DECEM
BRE 1999

 M. et Mme RAVIS
 M. et Mme RAVIS
ont la joie de vous annoncer
la naissance de leur bebe
 DESIRE
 DESIRE
le 31 DECEMBRE 1999
```

Deux lignes sur l'imprimante qui recopie la mémoire sans plus.

M.et Mme RAVIS

Une ligne en double hauteur sur l'écran

Un exemple ? Toute ressemblance avec des bébés étant réellement nés serait purement fortuite !

CHAPITRE VII

DESSINS EN HAUTE RÉSOLUTION

Après la basse résolution, voici la haute résolution, celle qui vous permettra les tracés les plus fins. Au cours de ce chapitre, on examinera quelques autres ordres Basic relatifs aux boucles, afin de terminer sur un festival de tracés et de couleurs.

Principaux thèmes abordés

Haute résolution	IF... THEN
HIRES	ELSE
TEXT	IF... THEN... ELSE
Ecran haute résolution	Branchement conditionnel
Points d'affichage	Branchement inconditionnel
CURSET	Opérateurs relationnels
CURMOV	AND
CIRCLE	OR
DRAW	NOT
PATTERN	Lecture d'une touche
Cercles à géométrie variable	KEY$
CHAR	Attributs
POINT	Clignotement
FILL	Lignes doublées
Lignes de couleurs aléatoires	CTRL/D
REPEAT UNTIL	Caractères en double hauteur

1. Placement du curseur

HAUTE RESOLUTION
HIRES

Nous allons abandonner la *basse résolution* (« LORES » vient de « *low resolution* », basse résolution) pour passer à la *haute résolution,* ce qui se dit en langage oricien HIRES, de « *high resolution* ». Frappez donc, en mode direct :

```
HIRES
```

Puis, faites un RETURN. L'écran noircit mais tout en bas subsistent trois lignes blanches.

Les trois lignes blanches inférieures sont destinées au texte, à vos ordres. Le reste de l'écran appartient au graphique.

Lorsque vous êtes en mode HIRES, la frappe de CTRL/L *ne vous ramène pas* au mode texte : elle efface uniquement les trois lignes du bas.

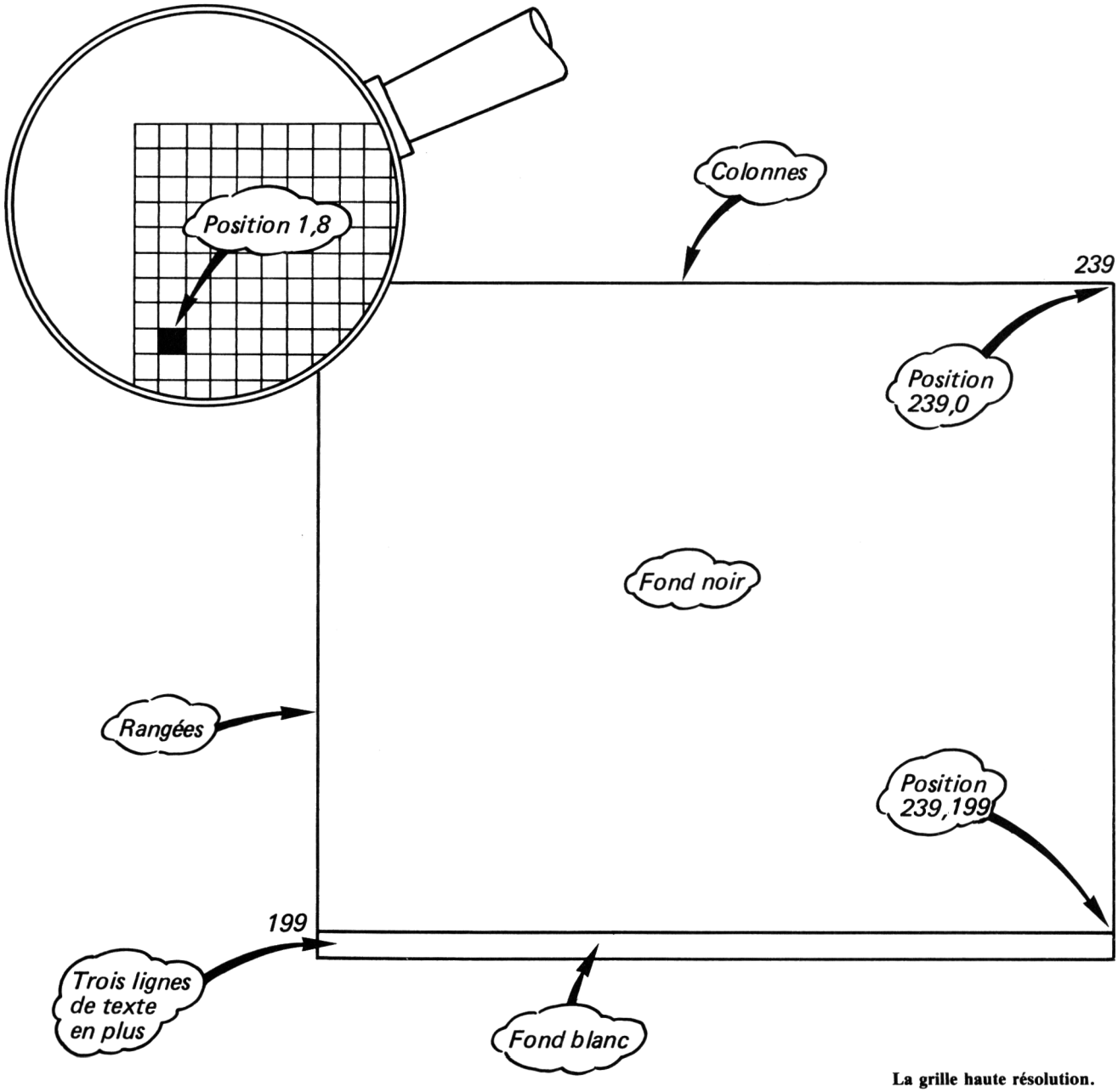
TEXT

Pour revenir au mode texte, frappez simplement TEXT en mode direct, puis RETURN.

ECRAN HAUTE RESOLUTION

POINTS D'AFFICHAGE

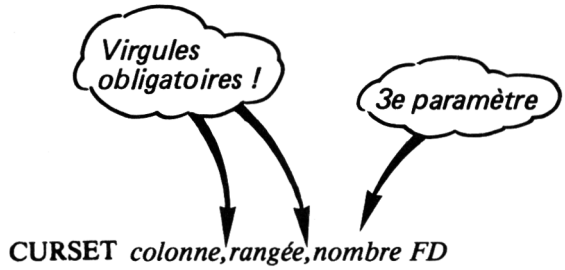
Vous pourriez passer aux modes graphiques basse résolution en frappant LORES Ø ou LORES 1.
Bien, mais que représente maintenant l'écran graphique ? En basse résolution, rappelez-vous, on l'a quadrillé selon une grille de 39 colonnes sur 27 rangées, soit 1 Ø53 positions au total. En mode *haute résolution,* la « définition » est bien supérieure car on dispose alors de 24Ø colonnes (numérotées de Ø à 239) et de 2ØØ rangées (numérotées de Ø à 199), soit un total de 48 ØØØ positions : reportez-vous à notre « grille », dessinée ici. Chaque position peut être assimilée à un point, tant elle est petite.



La grille haute résolution.

Il nous faut d'abord faire apparaître le curseur. Pour cela, on va le placer avec l'ordre CURSET, ce mot provenant de « cursor set », positionnement du curseur.

CURSET doit être suivi de trois valeurs, selon la syntaxe :

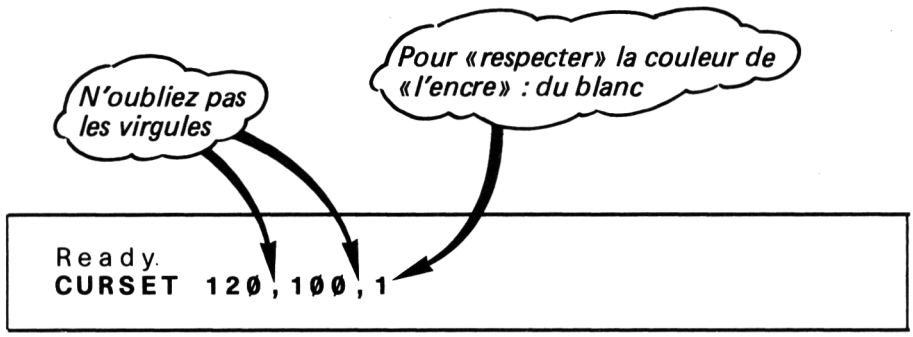


CURSET

Ce nombre est dit FD pour *fond/devant*, le *fond* étant tout simplement la couleur du fond (de la *page)* et le *devant,* la couleur de l'*encre* (de l'écriture). Les codes sont les suivants :

Nombre FD	Il affiche
Ø	Sur la couleur du fond (on peut) dire qu'on efface, alors)
1	Sur la couleur de l'encre
2	Inversion de couleur
3	Semble ne rien faire car n'affiche rien, mais met la position « colonne,rangée » en mémoire

Le plus évident consiste à expérimenter. Le « point » central de l'écran a pour coordonnées 12Ø,1ØØ. On va y mettre le curseur ; passez d'abord en mode HIRES, si ce n'est déjà fait, puis frappez ce texte qui viendra dans la fenêtre de texte de 3 lignes :



Au RETURN, un tout petit point blanc (couleur de « l'encre »), c'est le curseur, s'affiche au centre de l'écran. Maintenant, faites :

```
Ready
CURSET 120,100,2
```

Pour une inversion de couleur

Au RETURN et avec le 2 qui inverse la couleur, on place un curseur noir en 120,100 ce qui revient à « éteindre » le curseur précédent. Rallumez-le avec un CURSET 120,100,1 puis commandez :

```
Ready
CURSET 120,100,3
```

«Ne rien faire sur l'écran»

Au RETURN, rien ne se passe, *sur l'écran ;* en fait, le Basic s'est borné à mettre l'ordre *en mémoire.* Si une instruction suivante se réfère au curseur, et bien que celui-ci soit invisible, sa position sera prise en compte. Par contre, avec :

```
Ready
CURSET 120,100,0
```

Passons à la couleur du fond

et au RETURN, on affiche un curseur de la couleur du fond, ce qui revient à « éteindre » notre point blanc.
Jusqu'à présent, on a donné les coordonnées exactes de la position du curseur (120,100) ; c'est ce qu'on appelle des *valeurs absolues.* Or, on peut très bien déplacer le curseur en se référant à sa position précédente ; par exemple, on va le déplacer de 50 positions à droite et de 20 positions plus bas. Sa position *absolue* sera donc de 120 + 50 = 170 et de 100 + 20 = 120, mais grâce à l'ordre CURMOV (de « *cursor move* », déplacement du curseur) il suffira de donner des indications *relatives* selon la *même* syntaxe :

```
Ready
CURMOV 50,20,1
```

Ce sont des valeurs «relatives»

Au RETURN, un *second* point s'affiche en 170,120.

CURMOV

Nous avons bien dit « un second ». Chaque nouvel ordre CURSET ou CURMOV place un nouveau point, qu'on appelle « curseur » dans la mesure où on se réfère à lui pour l'opération suivante.

CURMOV accepte des valeurs négatives, pour des « marches arrières » :

```
Ready
CURMOV -100,-50,1
```

Déplacement à partir de la position précédente qui était 170, 120

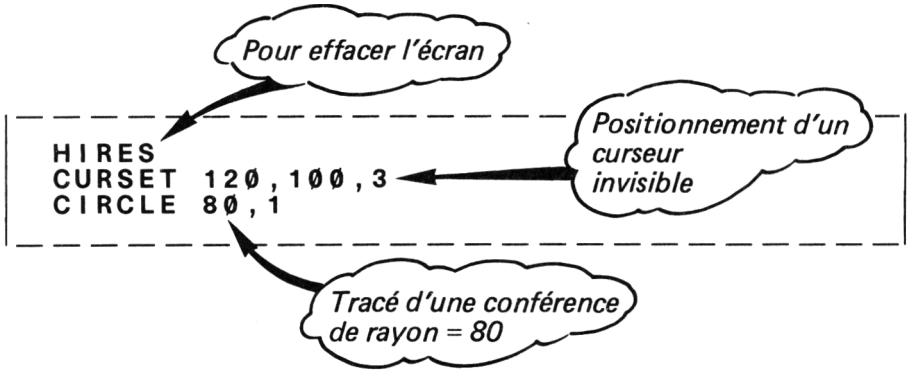
Au RETURN, un nouveau point s'allume en 170 – 100 = 70 et 120 – 50 = 70, ce qui correspond à la commande CURSET 70,70,1.

2. Tracés

La haute résolution dispose de commandes de tracés très efficaces ; par exemple, on peut tracer une circonférence avec *une seule* instruction CIRCLE. Nous allons vous montrez la série d'ordre à introduire à la file, mais rappelez-vous que vous n'en verrez qu'un à la fois sur l'affichage inférieur. Commencez par faire HIRES.

Même si vous êtes déjà en mode « haute résolution », HIRES servira à effacer l'écran.

Puis, placez un curseur « invisible » au centre, en 12Ø,1ØØ, grâce au troisième chiffre « 3 », comme on l'a vu. Enfin, faites CIRCLE en donnant le rayon et le « FD » comme d'habitude :



CIRCLE

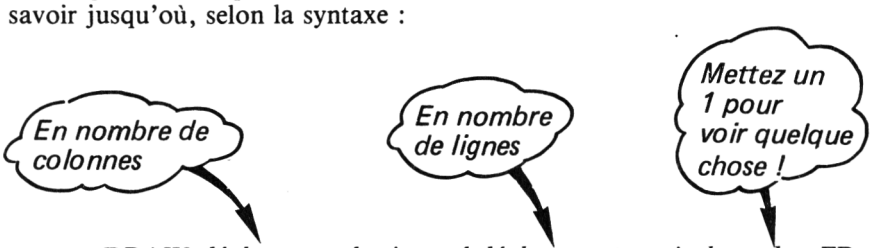
Au RETURN, une circonférence de rayon 8Ø s'inscrit autour du curseur fictif, au centre de l'écran. Notez la syntaxe de CIRCLE :

CIRCLE *rayon,nombre FD*

Notez qu'au lieu de CURSET 12Ø,1ØØ,3 vous auriez pu aussi bien écrire CURSET 12Ø,1ØØ,Ø avec le même résultat ici.

DRAW

Pour tirer un trait, employez DRAW (« tracez », en anglais) ; DRAW trace un trait à partir de la position du curseur et en se servant de *valeurs relatives* pour savoir jusqu'où, selon la syntaxe :



DRAW *déplacement horizontal,déplacement vertical,nombre FD*

Supposons que votre curseur se trouve toujours en 12Ø,1ØØ ; sinon, remettez-le à cette place.

La commande CIRCLE ne déplace pas le curseur. Par contre, DRAW le déplace et l'amène au bout du nouveau trait tiré.

Commandez alors :

```
DRAW 3Ø , 6Ø , 1
```

Mode «visible»

«Allez de 30 vers la droite ...»

«...De 60 vers le bas»

Et tracer un trait reliant ce nouveau point au curseur

Au RETURN, un trait oblique partant du centre de votre circonférence va s'inscrire, tel la grande aiguille d'une montre pointant 25 minutes.

Attention : maintenant, le curseur est au bout de votre trait, donc en 12Ø + 3Ø = 15Ø et en 1ØØ + 6Ø = 16Ø, donc en 15Ø,16Ø.

Vérifions-le en traçant un carré, à partir de 15Ø,16Ø, donc ; notre carré va s'accrocher au bout de l'aiguille de montre :

```
DRAW 5 , Ø , 1
DRAW Ø , 5 , 1
DRAW -5 , Ø , 1
DRAW Ø , -5 , 1
```

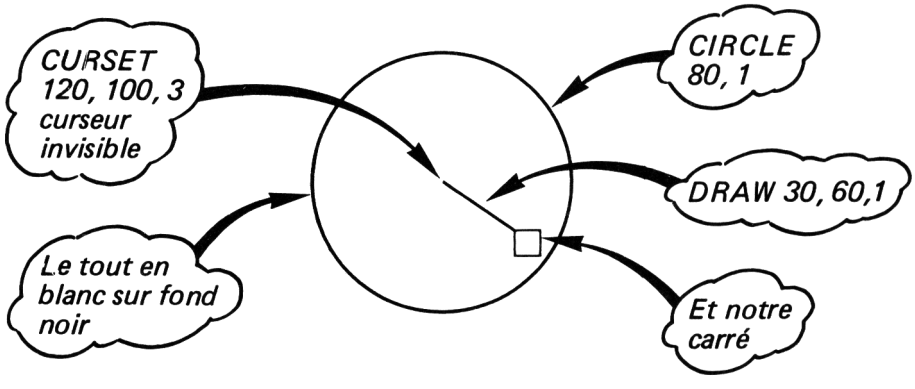
Côté horizontal supérieur

Côté vertical, à droite

Côté horizontal inférieur, vers la gauche

Côté vertical de gauche, vers le haut

Sur l'écran, vous avez maintenant ceci :

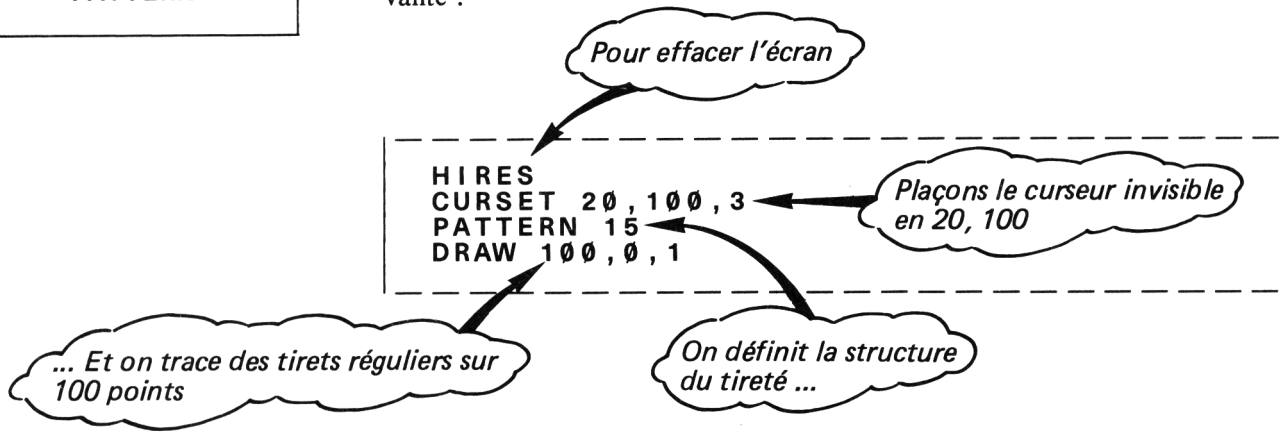


Appréciez la facilité des tracés et leur finesse ! Mais ce n'est pas tout : DRAW trace des traits continus mais, grâce à PATTERN, vous allez pouvoir tirer des traits discontinus. Il suffit de faire suivre PATTERN d'un nombre compris entre Ø et 255.

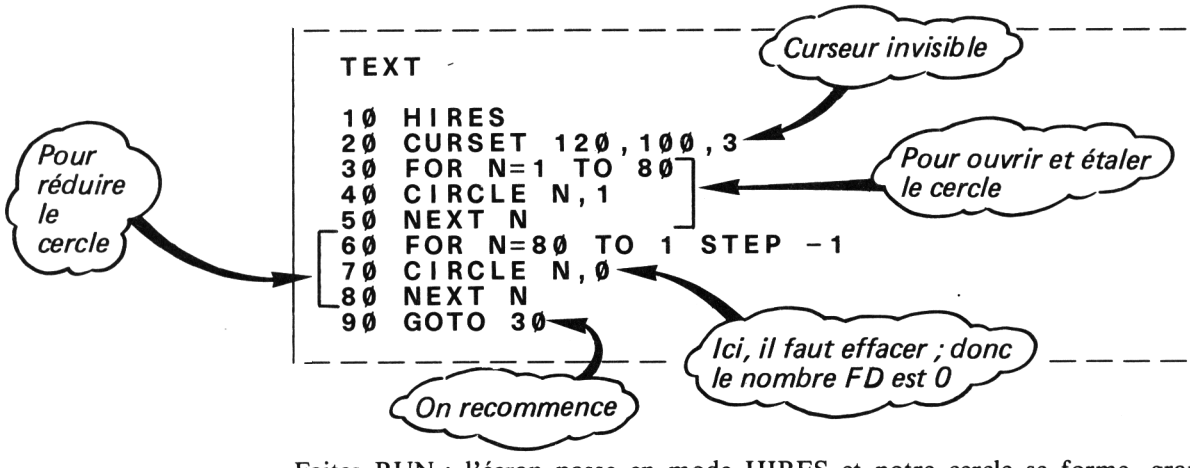
Pourquoi ces valeurs ? Si vous connaissez le binaire, vous savez que, sur 1 octet, on peut coder de Ø à 255. La structure du mot binaire (ses Ø et ses 1) définit l'aspect du trait à obtenir.

PATTERN

Mais sans connaître le binaire, faites quelques essais au hasard, ou plutôt, commencez par des pointillés réguliers avec PATTERN 15 ; la séquence est la suivante :



Essayons maintenant un programme qui va tracer un cercle de rayon s'accroissant continûment, puis décroissant jusqu'à disparaître, puis re-grandissant à nouveau, etc. Pour cela, on va rédiger les instructions en *mode texte*. Pour revenir à ce mode, frappez simplement TEXT, puis RETURN ; et entrez :



CERCLES A GEOMETRIE VARIABLE

Faites RUN : l'écran passe en mode HIRES et notre cercle se forme, grandit jusqu'à 8Ø, puis commence à diminuer, etc. C'est parce qu'il faut *effacer* que, pour le réduire, on a mis CIRCLE N,Ø à la ligne 7Ø. Arrêtez cela avec CTRL/C.
Essayons, maintenant, d'en faire autant avec des circonférences et non plus des cercles (la circonférence est le périmètre d'un cercle ; un cercle est une *surface* !). Pour cela, il faut *effacer* le cercle précédent après avoir affiché le suivant. Le programme est sensiblement le même que ci-dessus ; on a ajouté deux lignes d'effacement, en 45 et 75, et modifié les lignes 3Ø et 6Ø (bien que la modification de 6Ø soit totalement inutile : on aurait effacé une ligne inexistante, et après ? Mais c'est pour le principe) ; ces modifications sont dûes au fait que le Basic refuse de tracer un cercle de rayon Ø et considère que c'est une erreur ; il ne faut donc pas qu'un calcul de rayon donne zéro. Voici ce programme :

```
TEXT

10 HIRES
20 CURSET 120,100,3
30 FOR N=2 TO 80
40 CIRCLE N,1
45 CIRCLE N-1,0
50 NEXT N
60 FOR N=79 TO 1 STEP -1
70 CIRCLE N,1
75 CIRCLE N+1,0
80 NEXT N
90 GOTO 30
```

Pour effacer, en mode croissant, le cercle précédent : (N – 1)

Pour effacer, en mode décroissant : donc le cercle précédent était plus grand (N + 1)

Lancez-le, puis imaginez-en de plus savants. Par exemple, introduisez en 25 un :

```
25 PATTERN 1
```

Au RUN, vos cercles seront faits de points. Avec les grands rayons, vous voyez nettement un nouveau cercle se créer puis le précédent s'effacer.

CHAR

Sachez encore que vous pouvez aussi placer des caractères en HIRES, à l'emplacement du curseur, avec l'ordre CHAR (pour « *character* », mot anglais), mais sans le suffixe *dollars,* ici, (ce n'est pas un nom de variable chaîne !). La syntaxe est :

C'est le code ASCII du caractère, de 32 à 127

0 pour standard ou 1 pour caractère graphique

Comme précédemment

CHAR *code ASCII,standard ou graphique,nombre FD*

C'est facile à essayer. Faites CTRL/C si vos cercles sont toujours en évolution, puis :

Pour effacer l'écran

```
HIRES
CURSET 120,100,3
CHAR 83,0,1
```

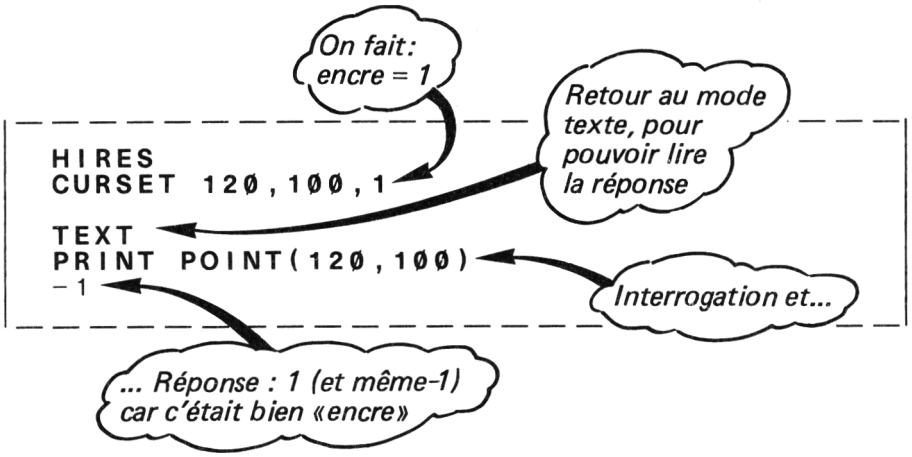
Placez le curseur encore une fois au centre

Pour afficher le caractère standard de code 83

Au dernier RETURN, le caractère de code 83, toujours un S, se place au centre de l'écran.

POINT

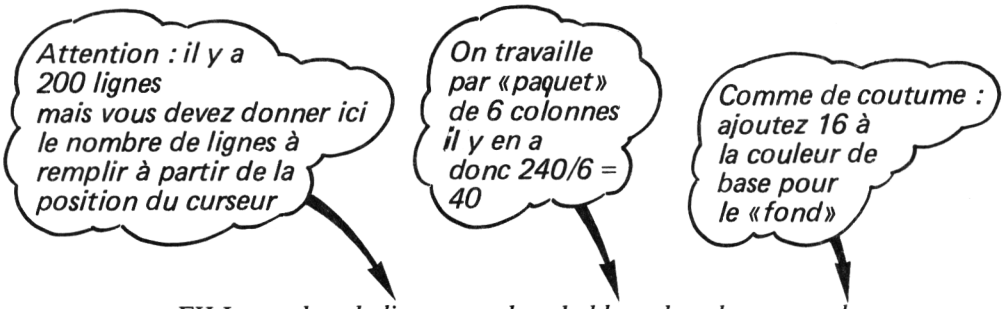
Enfin, vous pouvez demander au Basic de quelle « couleur » (*fond* ou *encre*) est un point quelconque, précisément avec l'ordre POINT. Il vous répondra par le nombre FD et donc par 0 pour *fond,* mais par – 1 pour *encre ;* encore faudra-t-il que vous reveniez en mode TEXT pour pouvoir lire la réponse. Cela donnera :



3. Couleurs en HIRES

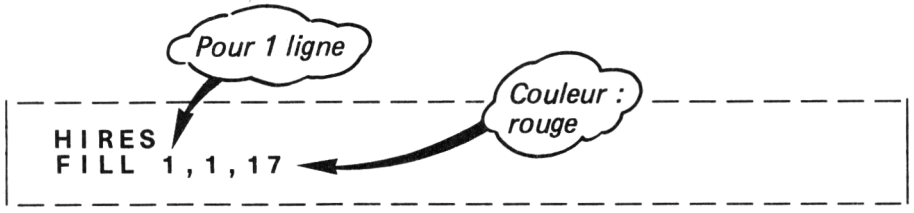
FILL

Mettre de la couleur en HIRES est un peu plus subtil. Notons, tout d'abord, que l'ordre est FILL (pour « remplir »), et qu'il doit être suivi par 3 paramètres :

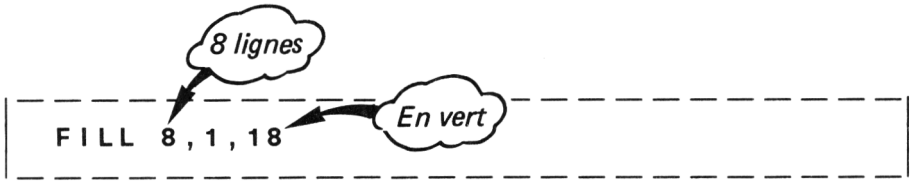


FILL *nombre de lignes,nombre de blocs de colonnes,couleur*

Ne vous préoccupez pas du *second* paramètre, pour l'instant, fixé à 1, et frappez :



Au RETURN, la ligne supérieure de l'écran passe en rouge. Pourquoi elle et elle seule ? Puisqu'on n'a pas indiqué de CURSET pour positionner le curseur, *celui-ci s'est mis d'office* en Ø,Ø ; d'autre part, le premier paramètre est 1, donc une seule ligne s'est teintée. Si vous faites maintenant :



Vous allez colorier en vert les 8 lignes suivantes.

Dès lors que vous mettez des lignes en couleur, le curseur se déplace de lui-même et passe à la ligne suivante.

Essayez de placer le curseur au début d'une ligne, par exemple la ligne 100, avec la position 0,100, pour placer du cyan (couleur 227) :

```
CURSET 0,100,1
FILL 15,1,22
```

Le curseur est visible

15 lignes

Au RETURN, 15 lignes se colorent en cyan (bleu-vert) à partir de la ligne 100. Bien, essayons, maintenant, de démarrer sur une colonne à mi-course :

```
CURSET 120,140,1
FILL 3,1,19
```

3 lignes

Jaune

Au RETURN, 3 lignes jaunes démarrent *à partir de la colonne 120*. Si vous refaites ensuite un :

```
FILL 7,1,20
```

Bleu

7 lignes en bleu vont se placer en-dessous, *toujours à partir de la colonne 120.*

Notez bien que c'est le curseur qui donne la position de la colonne de départ, pour FILL.

LIGNES DE COULEURS ALEATOIRES

Pour illustrer un peu les possibilités de couleurs de l'Oric, et faire une pause, entrez et exécutez ce programme qui remplit de couleurs aléatoires toutes les lignes, de 0 à 199. Il n'est pas nécessaire de positionner le curseur au départ puisqu'il se place automatiquement en 0,0 et que chaque ligne teintée le fait avancer d'une ligne :

```
TEXT
10 HIRES
20 FOR N=0 TO 199
30 C=RND(9)*8+16
40 FILL 1,1,C
50 NEXT N
```

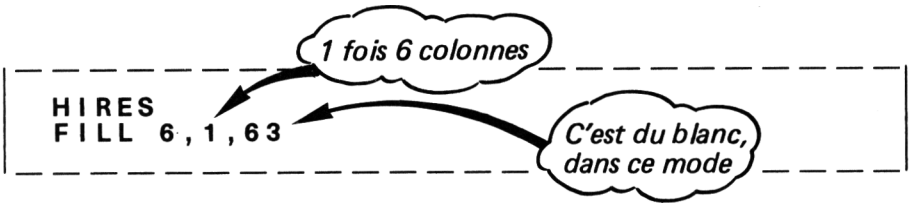
Pour balayer les 200 lignes

Choix aléatoire de la couleur

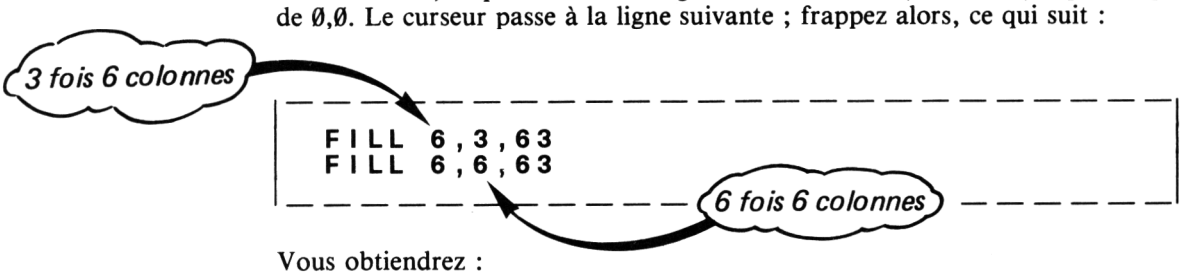
Faites RUN et vous obtiendrez une belle palette de lignes successives se suivant dans des couleurs placées au hasard.
Passons maintenant au paramètre *bloc de colonnes :* il agit par « paquets » de 6 colonnes d'un coup ; chaque ligne de 240 colonnes comporte donc 40 « paquets ».

Attention : ce paramètre ne fonctionne pas avec les couleurs ! Si vous le spécifiez, que ce soit de 1 à 40, c'est toujours toute la ligne qui va se colorier.

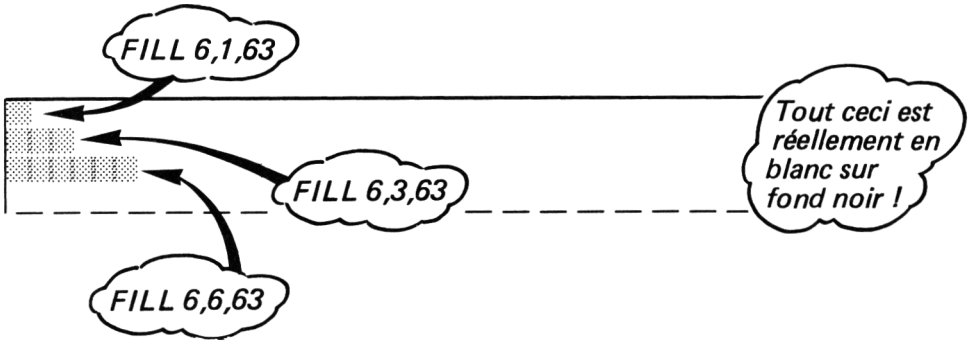
Par contre, il fonctionne avec des caractères spéciaux où le n° 63 représente un pavé blanc. Essayez :



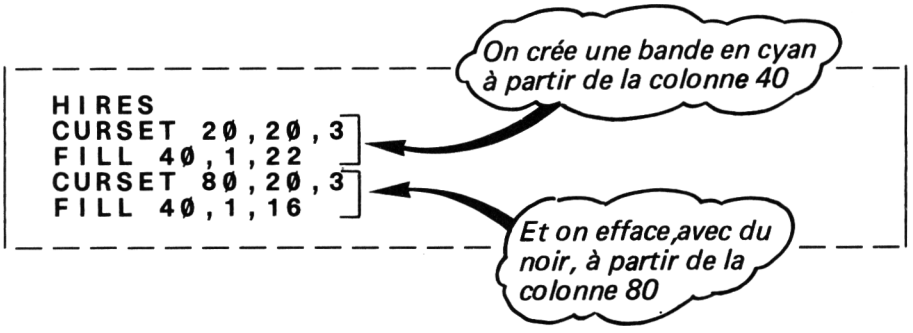
Au RETURN, un pavé *blanc* de 6 lignes sur 6 colonnes (1 « bloc ») se place à partir de Ø,Ø. Le curseur passe à la ligne suivante ; frappez alors, ce qui suit :



Vous obtiendrez :



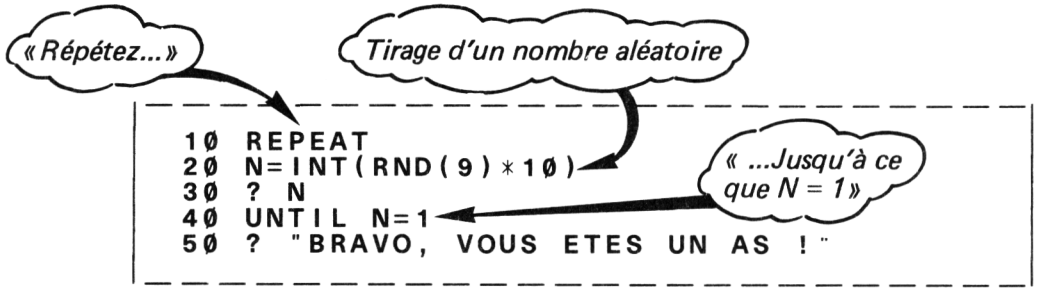
Au lieu de 63, vous pouvez placer d'autres codes. Par exemple, le code 45 vous donnera des verticales parallèles (FILL 9,12,45 par exemple), mais à l'évidence, c'est le code 63 qui sera le plus utile aux débutants.
Pour créer une surface en couleur, on va mettre toute la bande voulue en couleur, puis *effacer* (avec du noir) l'excédent. Par exemple, pour un rectangle :



Vous pouvez combiner des quantités de choses, maintenant, en HIRES, mais nous devons encore évoquer le 3e paramètre de FILL qui ne sert pas uniquement pour la couleur. Nous allons cependant, en « intermède », vous présenter quelques autres ordres importants du Basic « classique » qui vont nous servir. Pour cela, on va revenir au mode texte ; frappez donc TEXT, puis NEW.

4. Répétitions

On tire des nombres aléatoires entre Ø et 9 ; dès que l'on a tiré un as (un 1), on veut que le programme cesse. Comment réaliser cela ? Tout simplement en indiquant : *répéter jusqu'à ce que le nombre soit un as.* Le mot *répéter* s'écrit REPEAT, en anglais, alors que *jusqu'à* se dit UNTIL :

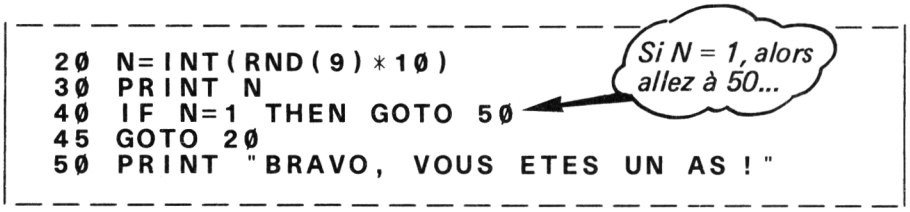


REPEAT UNTIL

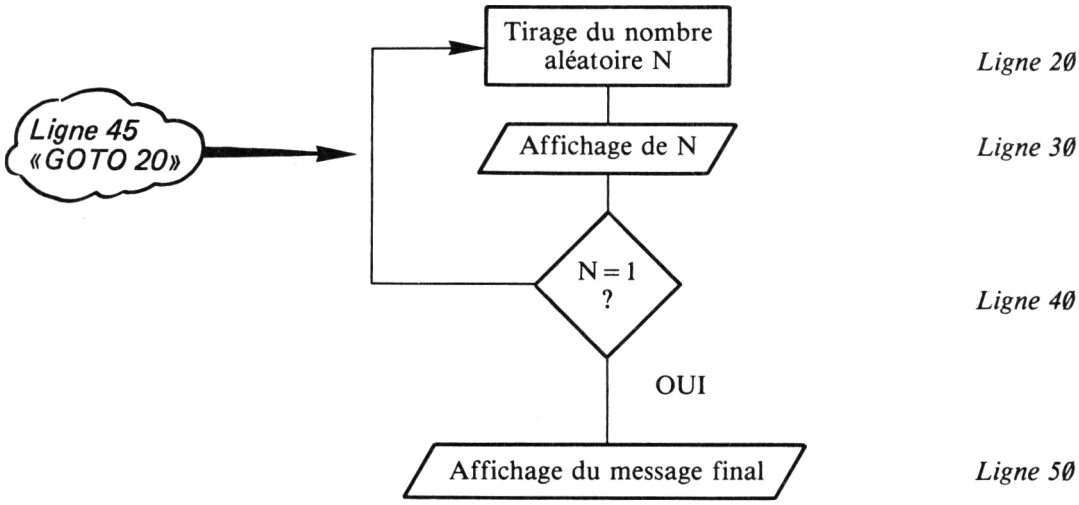
La ligne 10 REPEAT ordonne la répétition de *toute la séquence* incluse jusqu'à la ligne UNTIL (la ligne 40, ici), tant que N ne sort pas à 1. Donc, tant que N n'est pas 1, on va revenir de 40 à 10. Dès que N = 1, la boucle est terminée et le Basic passe en 50. Voici ce que donnerait un RUN :

IF... THEN

Il existe une autre façon de procéder, qui consiste à écrire « *si N = 1, alors écrire Bravo, vous êtes un as !* ». Le mot *si* se dit IF, *alors* se dit *then ;* c'est un peu plus compliqué ; commencez par supprimer la ligne 10 et conservez 20,30 et 50 ; puis, frappez 40 et 45, ce qui donne pour le programme complet :



Ici, la ligne 40 renvoie en 50 *à la condition que N = 1 ;* sinon, le renvoi est ignoré, et le Basic passe à l'instruction suivante qui est 45, avec le renvoi au début. Voici l'organigramme de ce programme :



Organigramme de la boucle IF... THEN

Faites un RUN : le résultat est rigoureusement le même. Notons que, dans la ligne 4Ø, on peut se permettre d'écrire tout simplement :

```
40 IF N=1 THEN 50
```

Le «GOTO» est implicite

ELSE

Bien mieux, on peut cumuler les lignes 4Ø et 45 en ajoutant, en 4Ø, une option supplémentaire : « *Si* N = 1 *alors* 5Ø, mais *sinon* aller en 2Ø ». Sachant que *sinon* s'écrit en anglais ELSE, le programme complet devient :

Si N = 1, on va en 50, sinon, on passe en 20

```
20 N=INT(RND(9)*10)
30 PRINT N
40 IF N=1 THEN 50 ELSE 20
50 PRINT "BRAVO, VOUS ETES UN AS!"
```

Un RUN donnera rigoureusement le même résultat.

BRANCHEMENT CONDITIONNEL, INCONDITIONNEL

L'ordre IF... THEN est appelé un « branchement conditionnel », ce qui semble évident : le branchement ne se fait que si la condition est remplie.
Par opposition, GOTO est appelé un « branchement inconditionnel ».

Dans ces programmes, la condition était l'*égalité,* avec N = 1. Mais il peut y en avoir d'autres, qui s'inspirent de l'arithmétique classique ou de la logique ; si vous les avez étudiées, ils n'offriront pas de mystère pour vous. En voici la liste :

OPERATEURS RELATIONNELS

OPERATEURS RELATIONNELS

Notation Basic	Signification	Notation classique
= < >	Egal Plus petit que Plus grand que	= < >
>= ou => <= ou =< <>	Plus grand ou égal Plus petit ou égal Différent de	⩾ ⩽ ≠
AND OR NOT	ET logique OU logique Négation logique	

AND

OR

NOT

5. Lecture d'une touche avec KEY$

LECTURE D'UNE TOUCHE
KEY$

L'ordre KEY$ (ce mot provient de « *key* », touche), ordonne au Basic de lire le clavier, au cours de l'exécution d'un programme ; celui-ci « ramène » le caractère correspondant à une touche précisément frappée à cet instant, sous forme d'une chaîne d'un seul caractère d'ailleurs. C'est ce qui explique la présence du symbole « dollar » au bout du mot *key*. La suite dépendra de ce que vous avez prévu. Dans le programme suivant on demande, par exemple, si la touche frappée est un Z et si oui, on affiche MERCI. Si non, on renvoie à ce même ordre de lecture du clavier avec KEY$, ce qui donne :

```
10 PRINT "FRAPPEZ SUR LE Z"
20 IF KEY$="Z" THEN PRINT "MERCI"
30 GOTO 20
```

« Lis le clavier : si la touche frappée est le Z, écris MERCI »

La ligne 10 affiche le message : *Frappez sur le Z.* A la ligne 20, on ordonne au Basic de lire sur le clavier (KEY$), ce qu'il fait ; s'il lit un Z, il affiche MERCI. Sinon, comme dans tout ordre IF... THEN, il passe à la ligne suivante qui le renvoie en 20 pour poursuivre la lecture du clavier. Au RUN, vous aurez :

Dès que vous avez frappé sur le Z

```
FRAPPEZ SUR LE Z
MERCI
```

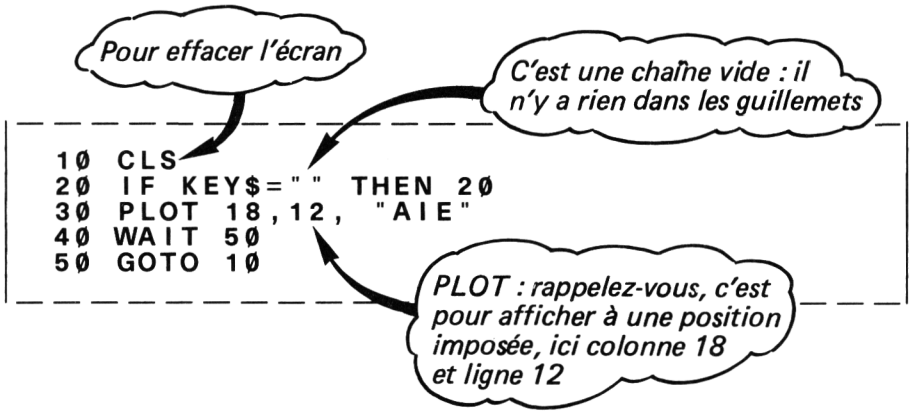
Si vous frappez sur le Z, le mot MERCI s'affiche. Mais ensuite, le programme continue d'attendre, et à chaque frappe du Z, il ré-affichera un nouveau MERCI. Afin de ne tenir compte que de la première frappe du Z, il faut compléter la ligne 20 avec un END, ce qui signifie que dès que l'on affiche un MERCI, le programme prend fin ; ce qui donne :

```
10 PRINT "FRAPPEZ SUR LE Z"
20 IF KEY$="Z" THEN PRINT "MERCI":END
30 GOTO 20
```

Rappelez-vous que le double-point est le séparateur officiel

Notez bien ce point essentiel : le Basic va lire le clavier à l'instant où il rencontre l'ordre KEY$. Il ramène le caractère (la chaîne de un caractère, plus exactement) correspondant à la touche qui est justement frappée à cet instant. Si aucune touche n'est frappée, il ramène une chaîne vide, (donc " ").

Voici un petit programme appliquant KEY$. Entrez-le, mais en le lisant, essayez de deviner ce qu'il va produire ; si vous ne trouvez pas, exécutez-le, et lisez la suite :



Au RUN, l'écran s'efface. Si vous frappez une touche, un « AIE » s'affiche un instant (WAIT 50) au centre de l'écran ! Faites CTRL/C pour stopper ce programme. Que s'est-il passé ? A la ligne 20, on commande la lecture du clavier avec KEY$; le Basic ne lit rien tant qu'une touche n'est pas frappée, ce qui correspond à une chaîne vide. Dans ce cas, la ligne 20 se renvoie à elle-même et le Basic va continuer à surveiller le clavier. Dès qu'une touche est frappée, la condition posée par le IF n'est plus remplie et dans ce cas, le THEN n'est pas exécuté : le Basic passe à la ligne suivante, qui commande l'affichage de AIE au centre de l'écran, puis on attend 0,5 seconde (ligne 40), et enfin on revient en 10 pour effacer l'écran et recommencer.

6. Les attributs

ATTRIBUTS

Après cette escapade, nous pouvons revenir à l'ordre FILL pour examiner son troisième paramètre qui fixe la couleur, *ou d'autres attributs.* Ce paramètre va de 0 à 127 inclus ; voici ses groupes d'actions :

Valeurs du paramètre	Action
0 à 7 inclus	C'est la couleur de l'*encre,* du texte
8 à 15 inclus	Agit sur l'affichage : voir plus loin
16 à 23 inclus	Couleur du fond, comme on l'a vu
24 à 31 inclus	Agit sur la « fréquence de balayage » de l'image : ne vous en servez pas !
32 à 127 inclus	Sert à la définition de motifs graphiques

Dans ce tableau, la ligne « obscure » est celle des valeurs de 8 à 15. Les fonctions sont les suivantes :

Paramètre	Action
8 9	Caractères alphanumériques du clavier standard Caractères graphiques (rappelez-vous du mode LORES 1)
10 11	Caractères standards, doublés (sur 2 lignes) Caractères graphiques, doublés (sur 2 lignes)
12 13	Clignotement, caractères alphanumériques Clignotement, mais caractères graphiques
14 15	Clignotement alphanumérique, doublé (2 lignes) Clignotement graphique, doublé (2 lignes)

CLIGNOTEMENT

Les deux nouveautés résident dans le *clignotement* et les *caractères doublés,* car on a déjà examiné le reste. Commençons par le clignotement : on va placer un cercle et le faire clignoter :

```
HIRES
CURSET 120,100,3
CIRCLE 70,1
CURSET 0,0,3
FILL 199,1,12
```

Création du cercle, comme on l'a déjà vu

On ramène le curseur en haut et à gauche ...

... Et on fait clignoter tout l'écran (sur 199 lignes)

Vous pouvez mettre ici 12 ou 13

Le second placement du curseur a été fait en 0,0 mais en réalité, il suffisait de le situer de façon qu'il englobe le cercle. Essayez d'autres valeurs, en modifiant ensuite le nombre de lignes à balayer (moins de 199, selon ce qu'il reste à partir du curseur).

LIGNES DOUBLEES

CTRL/D

Pour examiner les caractères doublés, on va procéder autrement : vous les obtiendrez soit avec le suffixe 10 ou 11, comme le montre le tableau ci-dessus, soit en frappant au clavier CTRL/D. Frappez CTRL/D puis, pour expérimenter, frappez FRERE JACQUES, ce qui donnera :

```
FRERE JACQUES
FRERE JACQUES
```

La ligne a été doublée

CARACTERES EN DOUBLE HAUTEUR

Refaites CTRL/D pour revenir au mode normal. En fait, ce doublage de ligne n'offre un réel intérêt que pour doubler des caractères de hauteur, soit en définissant des caractères spéciaux (ce que l'Oric vous permet, mais on ne vous entraînera pas jusque-là dans ce livre), soit à l'aide des caractères graphiques ou des commandes graphiques.

Exploiter tous ces ordres reste encore assez subtil, et ne peut venir que de l'expérience car, malheureusement, nous ne pouvons vous en montrer ici toutes les combinaisons. A moins de vous proposer un programme qui en mêle aléatoirement plusieurs : on va employer FILL pour remplir des rangées (de nombre aléatoire) avec un code – suffixe aléatoire qui pourra fournir de la couleur, du clignotement, des motifs graphiques. Ce programme offre, en outre, l'avantage d'incorporer un IF... THEN et un REPEAT UNTIL. Vous trouverez à la suite quelques explications nécessaires :

```
10 HIRES
20 REPEAT
30 A=RND(9)*128
40 IF A>23 AND A<32 THEN 30
50 C=RND(9)*240
60 R=RND(9)*100
70 CURSET C,R,3
80 FILL RND(9)*99+1,1,A
90 UNTIL KEY$<>""
```

Calcul de l'attribut

Placement du curseur

Remplissage en fonction de A

Curieux ?

La ligne 2Ø ordonne la répétition de toute la séquence allant jusqu'à 9Ø. Le UNTIL KEY$<> " " signifie *jusqu'à ce qu'une touche soit frappée !* Décomposons : <> signifie différent de, et " " est une chaîne vide ; or, dès qu'une touche est frappée, elle se traduit par un caractère que sera « A », ou « B », ou l'espace donc, une chaîne non vide ; donc, à ce moment, le programme stoppera.

En 3Ø, on calcule aléatoirement la valeur de l'attribut A, qui viendra au bout de FILL. Mais attention : si vous considérez notre tableau précédent, vous constaterez que les valeurs 24 à 31 sont interdites. Par conséquent, on cherchera, à la ligne 4Ø, si A ne tombe pas dans cette gamme et si oui, on repart en 3Ø tirer une autre valeur. Le calcul se fait ainsi : *Si A est supérieur à 23 mais aussi* (c'est le ET logique, qui s'écrit AND) *si A est inférieur à 32* (donc, s'il est compris entre 24 et 31), *aller en 3Ø.*

Puis, en 5Ø et 6Ø, on calcule aléatoirement le positionnement du curseur, colonne et rangée sans « sortir » de l'écran, ce que le Basic identifierait à une erreur, et l'on place ce curseur (invisible) en 7Ø. Remarquez que les calculs de RND ne font pas appel à l'entier de la valeur obtenue : les fonctions utilisant ces valeurs s'en chargent d'office.

Enfin, en 8Ø on commande le FILL à partir, donc, d'un curseur placé aléatoirement, sur un nombre de rangées aléatoires (de Ø à 99, qui s'ajoutent donc à R calculé auparavant) et avec un paramètre A aléatoire. Il est difficile de faire davantage.

Lancez ce programme : vous allez voir des plaques de couleurs s'afficher, des caractères graphiques apparaître *en colonne* surtout car le FILL est doté d'un 1 en second paramètre, des clignotements se manifester... Le jeu changera constamment, jusqu'à ce que vous frappiez sur une touche quelconque (résultat de la ligne 9Ø). L'image se figera alors, mais les clignotements se poursuivront.

QUESTIONS SUR LE CHAPITRE VII

Vous en connaissez maintenant le principe !

1. Dans le mode haute résolution, la définition de l'écran est de :
   ☐ 200 rangées sur 240 colonnes
   ☐ 24 rangées sur 80 colonnes

2. L'ordre CURSET sert à :
   ☐ Tracer des traits en blanc sur fond noir
   ☐ Placer le curseur

3. CURMOV positionne le curseur :
   ☐ Selon des coordonnées absolues (colonne,rangée)
   ☐ Par rapport à sa position antérieure

4. L'ordre CIRCLE doit être accompagné de paramètres précisant :
   ☐ son centre, son rayon, et l'inévitable nombre FD
   ☐ Son rayon et le nombre FD

5. DRAW tire un trait entre :
   ☐ Deux points dont il faut donner les coordonnées en valeur absolue
   ☐ La position actuelle du curseur et un point dont on donne les coordonnées en valeurs relatives à cette position

6. L'ordre FILL sert :
   ☐ Uniquement à mettre de la couleur en HIRES
   ☐ A mettre de la couleur, mais aussi à exécuter d'autres commandes selon l'attribut (clignotement, motifs graphiques...)

7. REPEAT UNTIL ordonne :
   ☐La répétition d'une séquence tant que la condition posée n'est pas réalisée
   ☐ Une temporisation

8. Si N = 7, qu'affichera le programme suivant :

```
50 IF N>8 THEN 70
60 ? "BONJOUR":END
70 ? "AU REVOIR":END
```

   ☐ BONJOUR
   ☐ AU REVOIR

9. L'ordre KEY$, pour lire une touche :
   ☐ Stoppe l'exécution du programme pour attendre la touche
   ☐ Lit ce qui est frappé à la volée, au moment où le Basic rencontre cet ordre, mais n'arrête pas le programme

Réponses page 176

19

CHRONOMÈTRE DIGITAL PRÉCIS

STR$ VAL DEEK DOKE PEEK POKE

Pour un chronométrage précis, pourquoi ne pas se servir de l'horloge qui cadence les opérations élémentaires de l'Oric !

Elle est contrôlée par un quartz et doit donc donner de bons résultats. Le moyen le plus facile pour l'utiliser est l'instruction WAIT et c'est ce que nous avons déjà fait. Mais pour afficher il faut au minimum une instruction PRINT, ce qui va fausser le chronométrage. Une autre méthode consiste à interroger un ou deux octets de mémoire vive qui sont liés à l'horloge. C'est ce que nous allons faire :

1 — "Gratouiller" les octets 630 et 631 pour fixer l'heure de départ.
2 — Observer la variation du contenu de ces 2 octets.
3 — Afficher ce contenu.

De cette façon, on obtient directement un comptage à rebours au centième de seconde, sur une durée maximum de 65 535 centièmes de secondes, soit près de 11 minutes.

```
10 CLS
20 INPUT"HEURE DE DEPART 1/100 S";HD
30 DOKE 630,HD
40 T=DEEK(630)
50 PLOT 10,3,STR$(T)
60 GOTO 40
```

REMARQUES

30 — point de départ du compte à rebours

40 — un instant parmi d'autres

50 — affichage

Fournissez comme heure de départ, 1 000, ce qui représente 10 secondes. On observe un défilement rapide des 2 chiffres de droite, et celui de gauche qui bat la seconde. Mais nous n'avons rien prévu pour arrêter le chrono, et ce qui se passe en-dessous de zéro peut surprendre !

Arrivé à zéro les deux octets reprennent chacun la valeur la plus grande : 255, ce qui donne à notre compteur une nouvelle jeunesse : $(255 \times 256) + 255$ soit 65 535. Et c'est reparti pour 655 secondes 35/100 !

Les chiffres s'affichent en vert, et pourtant on ne l'a pas demandé !

T est un nombre positif, et à la place du signe + se trouve un caractère invisible, dont le code est 2.

Or c'est justement le code du vert comme couleur d'encre.

Au bout de 3 s 67/100

```
HEURE DE DEPART 1/100 S? 1000

613
```

DEEK
DOKE

Nous avons employé deux fonctions aux noms barbares qui ne sont pas du Basic courant : DEEK et DOKE.

Elles concernent toutes deux une paire d'octets voisins, DEEK pour lire et DOKE pour écrire.

A la ligne 30 on met la valeur de HD, dans les octets 630 et 631.
Si HD vaut 1 000, l'octet 631 recevra 3 et l'octet 630 : 232, car $3 \times 256 + 232 = 1\ 000$.

Puis grâce à DEEK (630) on lit ce qui se trouve dans ces 2 octets et le résultat est : (contenu de 631)*256 + (contenu de 630).

PEEK
POKE

Leur rôle est aussi de lire (PEEK) et d'écrire (POKE), mais un octet à la fois. PEEK (630) fournirait la partie basse du compteur et PEEK(631) la partie haute. POKE 630,0 ne remettrait à zéro que la partie basse de notre horloge

Chronomètre digital avec arrêts provisoires et temps intermédiaire

```
D:DEPART
A:ARRET
S:RATTRAPANTE
Z:REMISE A ZERO
F:FIN
```

Le menu du prochain programme

Ce sont les 5 fonctions d'un chronomètre digital enrichi, avec affichage séparé des secondes et des centièmes.

Ce programme comporte 5 boucles dans lesquelles on évolue grâce aux 5 lettres DASZF, et nous avons voulu construire ces boucles comme des blocs d'instruction dont voici la liste :

Attente → Répéter
  jusqu'à ce qu'on frappe D
On part à zéro → Répéter
  départ
  Répéter
Le chrono tourne →
    Répéter
      lecture et affichage du temps
      lecture du temps
      jusqu'à ce qu'on frappe A
Redémarrage avec ou sans rattrapage →
    Répéter
      lecture du clavier
      jusqu'à ce qu'on frappe D ou F ou S ou Z
    si on a frappé D, on repart où on s'était arrêté jusqu'à ce qu'on ait frappé autre chose que D et S
Redémarrage à zéro →
  jusqu'à ce qu'on ait frappé F
Fin

```
10 CLS:PAPER 7
20 PLOT2,4,"D:DEPART"
30 PLOT2,5,"A:ARRET"
40 PLOT2,6,"S:RATTRAPANTE"
50 PLOT2,7,"Z:REMISE A ZERO"
60 PLOT2,8,"F:FIN"
70 PLOT 1,2,17
80 PLOT 10,2,23
90 REPEAT:UNTIL KEY$="D"
110 DOKE 630,65535
100 REPEAT
110 DOKE 630,65535
120 REPEAT
130 REPEAT
140 T=DEEK(630)
150 T$="   "+STR$(65535-T)
160 L=LEN(T$)
170 S$=MID$(T$,L-4,3)
180 C$=RIGHT$(T$,2)
190 PLOT 2,2,S$
200 PLOT 5,2,":"+C$
210 T=DEEK(630)
220 UNTIL KEY$="A"
230 REPEAT
240 K$=KEY$
250 UNTIL K$="D" OR K$="F" OR K$="S" O
R K$="Z"
260 IF K$="D" THEN DOKE 630,T
270 UNTIL K$<>"D" AND K$<>"S"
280 UNTIL K$="F"
```

REMARQUES

20 à 60 — le mode d'emploi

70 à 80 — un beau cadre rouge

90 — attente : à "D" on part

100 à 280 — boucle pour des chronométrages successifs avec remise à zéro

110 — remise à zéro

120 à 270 — boucle pour redémarrage sans remise à zéro

130 à 230 — boucles de chronométrage jusqu'à ce qu'on tape D

140 — T : le contenu du "décompteur"

150 — T$: la chaîne contenant 65 535 – T pour compter à partir de zéro et non pas décompter, et 3 espaces à gauche pour cadrer

160 — la taille de T$

170 — les secondes

180 — les centièmes

190 — affichage des secondes

200 — affichage d'un : et des centièmes

210 — mise en mémoire du temps qui servira si on arrête le chrono (arrêt de jeu)

230 à 250 — boucle d'attente jusqu'à ce qu'on relance le chrono avecw D, S, Z, ou F

240 — observation du clavier : si une touche est frappée, son code va dans R$

260 — si on tapé D, on repart sans rattrapage

L'emploi de la structure REPEAT... UNTIL a l'avantage de ne pas faire appel aux numéros de lignes comme le fait GOTO...

Ceci est commode en particulier si on veut placer une portion de programme dans une autre, comme ce sera le cas si vous souhaitez réutiliser ce programme comme auxiliaire dans vos programmes nécessitant un chronométrage.

20

CARACTÈRES ACCENTUÉS

PEEK POKE READ DATA

On peut redessiner n'importe quel caractère du clavier en respectant une convention fixée par le constructeur. Nous allons le faire pour combler une lacune : l'absence de caractères accentués. Il y a deux possibilités principales : modifier certains caractères standards ou toucher au jeu de caractères semi-graphiques. Laissons de côté ces graphiques carrés pour remplacer plutôt certains caractères du jeu standard peu employés dans un texte.

Caractère existant	Code ASCII	Nouveau caractère
@	64	à
#	35	â
^	94	é
£	95	è
¦	124	ê
\	92	ë
[	91	î
]	93	ï
{	123	ô
}	125	û
<	60	ù
>	62	ç

```
10 CLS:PRINT:PRINT"CREATION DE CARACTE
RES ACCENTUES"
20 GOSUB 120
30 PRINT:PRINT"VOICI LES CARACTERES OB
TENUS :"
40 PRINT
50 FOR C=1 TO 12
60 PRINT CHR$(A(C));" ";
70 NEXT C
80 PRINT
90 :
100 END
```

Un espace

REMARQUES

20 — appel du sous-programme de création

50 à 70 — boucle d'affichage des caractères obtenus

```
110 :
120 REM-------------------------
130 DIM A(12)
140 DATA 64,35,94,95,124,92,91,93,123,
125,60,62
150 FOR C=1 TO 12
160 READ A(C):NEXT C
170 :
180 DATA 16,8,28,2,30,34,30,0
190 DATA 8,20,28,2,30,34,30,0
200 DATA 4,8,28,34,60,32,30,0
210 DATA 16,8,28,34,60,32,30,0
220 DATA 8,20,28,34,60,32,30,0
230 DATA 20,0,28,34,60,32,30,0
240 DATA 8,20,0,24,8,8,28,0
250 DATA 0,20,0,24,8,8,28,0
260 DATA 8,20,28,34,34,34,28,0
270 DATA 8,20,0,34,34,34,28,0
280 DATA 16,8,0,34,34,34,28,0
290 DATA 0,0,14,16,16,16,14,4
300 FOR C=1 TO 12
310 FOR L=0 TO 7
320 READ D
330 POKE 46080+A(C)*8+L,D
340 NEXT L
350 NEXT C
360 :
370 RETURN
```

REMARQUES

120 à 370 — sous-programme de création des nouveaux caractères

130 — réservation pour le fichier des codes

140 — les codes des caractères à redessiner

150 à 160 — boucle de lecture de ces codes

180 à 290 — le "dessin" des 12 caractères

300 à 350 — boucle pour chaque caractère

310 à 340 — boucle pour les 8 lignes d'un caractère

320 — lecture pour une ligne

330 — modification pour cette ligne

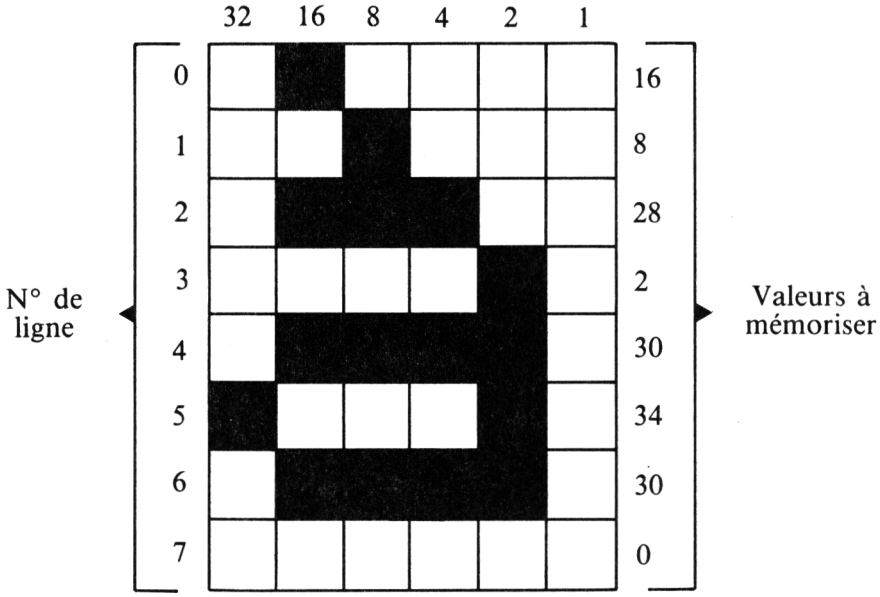
L'exécution du programme donne ceci :

```
CREATION DE CARACTERES ACCENTUES

VOICI LES CARACTERES OBTENUS :

à â é è ê ë î ï ô û ù ç

Ready
```

et à partir de maintenant si vous tapez @ c'est un *à* qui s'affiche, pour # vous obtiendrez *â*...

Ceci veut dire que vous pourrez créer, stocker, comparer et afficher du texte contenant ces 12 caractères greffés.

Pour illustrer la démarche suivie, prenons le cas du caractère *à* que nous dessinerons en respectant la grille imposée de 8 lignes et 6 colonnes :



Ce dessin est mis en mémoire aux emplacements voulus grâce à

```
POKE 46080+A(C)*8,16
.
.
.
POKE 46080+A(C)*8+7,0
```

on a choisi pour *à* le code 64 ; ce caractère sera donc obtenu grâce à la touche .

Ainsi pour écrire bérêt il faudra taper b∧r!t

Si vous utilisez ceci avec des programmes où figurent des caractères modifiés, le nouveau symbole sera affiché, mais le code ASCII restant le même, Basic s'en moque :

un <sera noté ù mais correspondra bien à "inférieur à" !

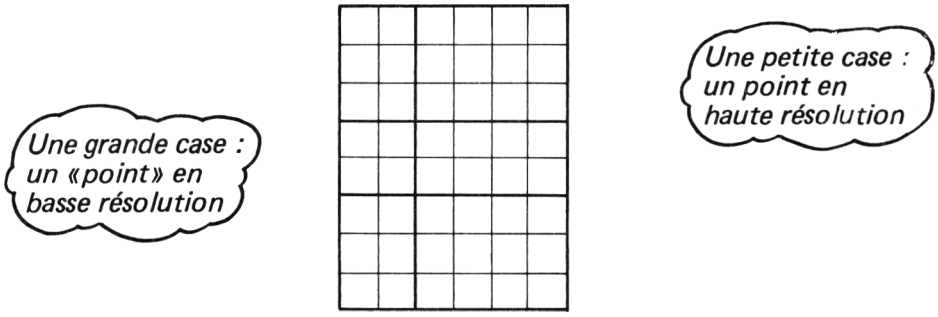
Pour revenir au jeu de caractères "normaux", utilisez le bouton de réinitilisation ou tapez CALL 555.

21

LES CARACTÈRES SEMI-GRAPHIQUES À LA LOUPE

HIRES CURSET CHAR DRAW

En fin d'ouvrage, on trouvera une liste des caractères du jeu alterné (numéro 1) dont le code est compris entre 32 et 95. Voici un programme qui affiche le caractère choisi et reproduit les points qui le compose dans une grille très agrandie formée de 8 cases dans le sens de la longueur et 6 cases dans la largeur.



Les caractères obtenus seront composés de cases noires et blanches formant ce quadrillage :

```
10 HIRES
20 FOR X=90 TO 126 STEP 6
30 CURSET X,90,1:DRAW 0,48,1
40 NEXT X
50 FOR Y=90 TO 138 STEP 6
60 CURSET 90,Y,1:DRAW 36,0,1
70 NEXT Y
80 :
90 FOR CT=32 TO 95
100 CURSET 10,10,0
110 CHAR CT,1,1
120 PRINT"CODE :":PRINT CT
130 :
140 FOR Y=10 TO 17
150 FOR X=10 TO 15
160 P=-POINT(X,Y)
170 CURSET X*6+31,Y*6+30,3
180 CHAR 43,0,P
190 NEXT X
200 NEXT Y
210 :
220 CURSET 10,10,3
230 CHAR CT,1,2
240 NEXT CT
250 :
260 END
```

REMARQUES

20 à 70 — dessin de la grile

20 à 40 — les barres verticales

50 à 70 — puis les barres horizontales

90 à 240 — boucle pour chaque caractère

100, 110 — affichage du caractère en haut à gauche

120 — son code est écrit dans les 3 lignes du bas

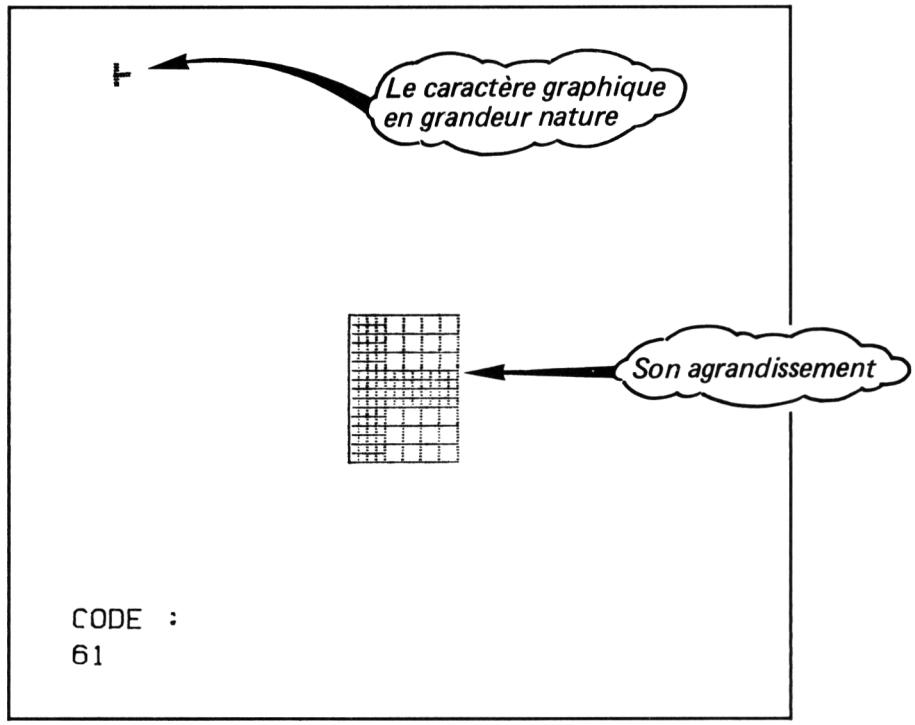
140 à 200 — boucles de remplissage de la grille

160 — P vaut 1 pour un point allumé, zéro sinon

170, 180 — marquage d'une croix lorsque P vaut 1

220, 230 — effaçage du caractère en haut à gauche avant de passer au suivant pour éviter une superposition

exemple : voici la forme du caractère dont le code est 70.



Passez-les systématiquement en revue (codes 32 à 95) et vous comprendrez mieux comment on passe d'un caractère au suivant.

Au-delà de 95 le spectacle est moins varié : on retrouve par exemple le même caractère 16 fois de suite ! Puis la série recommence à partir de 160.

22

CRÉATION DE MOTIFS

HIRES CURSET CHAR DRAW DEF FN

Le clavier de l'Oric comporte les symboles usuels tels que lettres de l'alphabet, chiffres et signes de ponctuation, mais on peut encore créer d'autres symboles qui occuperont la même place sur l'écran qu'une lettre ou un chiffre, et pourront être manipulés de la même façon. La création de tels motifs est décrite dans le manuel fourni avec le micro-ordinateur, et vous pouvez en trouver plusieurs exemples dans le présent ouvrage (graphisme, jeux d'animation...).

La démarche à suivre n'est pas très difficile, mais demande un soin attentif et une certaine habitude. Voici un programme qui vous facilitera le travail : vous dessinez le motif souhaité en noircissant les cases d'une grille agrandie, et le résultat obtenu est une instruction DATA décrivant le motif. Au fur et à mesure que vous le dessinez sur la grile, le futur motif apparaît sur l'écran en grandeur nature. Ceci permet des retouches afin de parvenir à l'effet recherché. Il suffira ensuite de prévoir dans votre programme la lecture des informations contenues dans l'instruction DATA pour disposer du motif comme s'il avait toujours existé !

```
10 HIRES
20 FOR X=90 TO 126 STEP 6
30 CURSET X,90,1:DRAW 0,48,1
40 NEXT X
50 FOR Y=90 TO 138 STEP 6
60 CURSET 90,Y,1:DRAW 36,0,1
70 NEXT Y
80 :
90 INPUT"JEU";J
100 INPUT"CODE";CT
110 CURSET 10,10,0
120 CHAR CT,J,1
130 :
140 FOR Y=10 TO 17
150 FOR X=10 TO 15
160 IF POINT(X,Y)=-1 THEN CURSET X*6+3
1,Y*6+30,3:CHAR 43,0,1
170 NEXT X
180 NEXT Y
190 :
200 DEF FNC(C)=C*6+85
210 DEF FNL(L)=L*6+84
220 L=1:C=1
230 :
```

REMARQUES

10 à 70 — tracé de la grille

20 à 40 — boucle de tracé des traits verticaux

50 à 70 — boucle de tracé des traits horizontaux

90 — choix du jeu de caractères 0 ou 1

100 — code ASCII du caractère à modifier

110, 120 — affichage de ce caractère tel qu'il est

140 à 180 — boucle d'analyse de caractère et représentation avant agrandissement dans la grille

160 — pour chaque point formant le caractère, on marque une croix (+) dans la case correspondante

200 et 210 — définition des fonctions de position du curseur dans la grille

220 — position de départ : en haut à gauche

```
240 REPEAT
250 A=POINT(C+9,L+9)
260 CURSET FNC(C),FNL(L),3
270 CHAR 43,0,-NOT A
280 WAIT 5
290 GOSUB 530
300 CHAR 43,0,-A
310 WAIT 5
320 UNTIL T$=CHR$(13)
330 :
340 DIM F(8)
350 FOR L=1 TO 8
360 F(L)=0
370 FOR C=1 TO 6
380 F(L)=2*F(L)-POINT(C+9,L+9)
390 NEXT C
400 NEXT L
410 :
420 TEXT
430 PRINT:PRINT"VOICI L'INSTRUCTION DA
TA NECESSAIRE   POUR OBTENIR LE MOTIF
:"
440 PRINT:PRINT"DATA ";
450 FOR L=1 TO 8
460 PRINT F(L);",";
470 POKE 46080+1024*J+CT*8+L-1,F(L)
480 NEXT L
490 PRINT CHR$(8);" "
500 PRINT:PRINT:PRINT"MOTIF OBTENU :";
510 IF J=0 THEN PRINT" ";CHR$(CT) ELSE
 PRINT CHR$(27);"I";CHR$(CT)
520 END
530 :
540 REM----------------------
550 T$=KEY$
560 IF T$<>"" THEN CODE=ASC(T$) ELSE R
ETURN
570 IF T$="0" THEN CURSET C+9,L+9,0:CU
RSET FNC(C),FNL(L),3
580 IF T$="1" THEN CURSET C+9,L+9,1:CU
RSET FNC(C),FNL(L),3
590 IF CODE<8 OR CODE>11 THEN RETURN
600 C=C+(CODE=8)-(CODE=9)
610 L=L-(CODE=10)+(CODE=11)
620 IF C<1 THEN C=1
630 IF C>6 THEN C=6
640 IF L<1 THEN L=1
650 IF L>8 THEN L=8
660 RETURN
```

REMARQUES

240 à 320 — répétition jusqu'à ce qu'on tape ''return''

250 — mémorisation de l'état de la case allumée ou éteinte

260 — on place le curseur dans la case (coin supérieur gauche) sans modifier ce qui s'y trouve

270 à 310 — clignotement du signe +

270 — effacement du +

280 — temporisation

290 — observation du clavier et modification d'une case : utilisation du sous-programme

300 — affichage du +

310 — temporisation

340 à 400 — calcul des valeurs définissant le nouveau motif

340 — réservation pour le fichier des codes de chaque ligne

350 à 400 — boucle sur les lignes

360 — remise à zéro de la valeur caractéristique d'une ligne

370 à 390 — boucle sur les 6 cases d'une ligne

380 — calcul progressif de la valeur

420 à 510 — résultat : ce qu'il faut faire pour recréer le motif plus tard

440 à 480 — écriture de l'instructin DATA et mise en mémoire des valeurs définissant le nouveau motif

REMARQUES

550 à 660 — sous-programme de déplacement dans la grille et de modification de celle-ci

550 — mémorisation d'une frappe de touche

560 — si on a appuyé sur une touche, CODE contient son code ASCII, sinon : fin du sous-programme

570 — cas du zéro : effaçage du point dans le petit caractère et de la croix dans la grille

580 — cas du 1 : allumage du point et d'une croix

590 — fin du sous-programme pour les codes autres que ceux des 4 flèches

600 — nouvelle position horizontale : c diminue de 1 si on a tapé ←, c augmente de 1 pour →, c ne change pas pour les 2 autres flèches

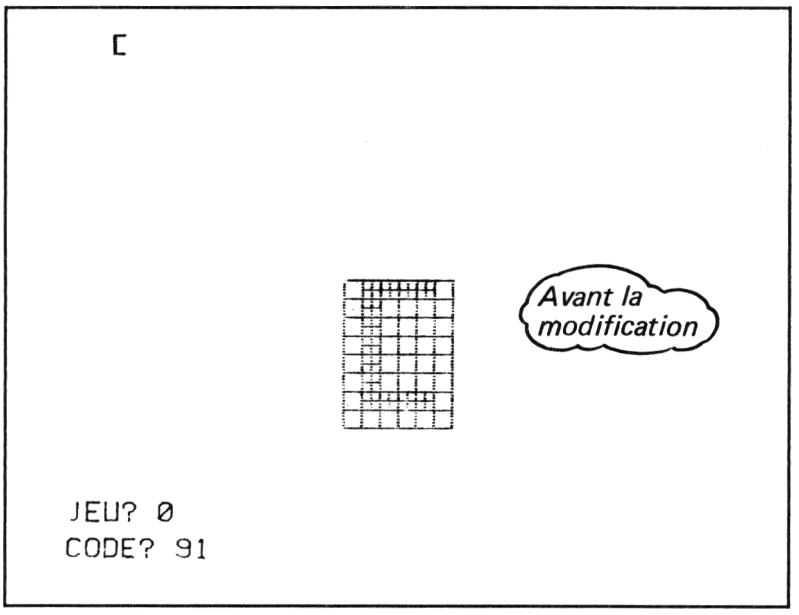
610 — même démarche pour la nouvelle position verticale : ↑ L diminue de 1, ↓ L augmente de 1, sinon L ne change pas

620 à 650 — blocage à gauche, à droite, en haut et en bas

660 — fin du sous-programme

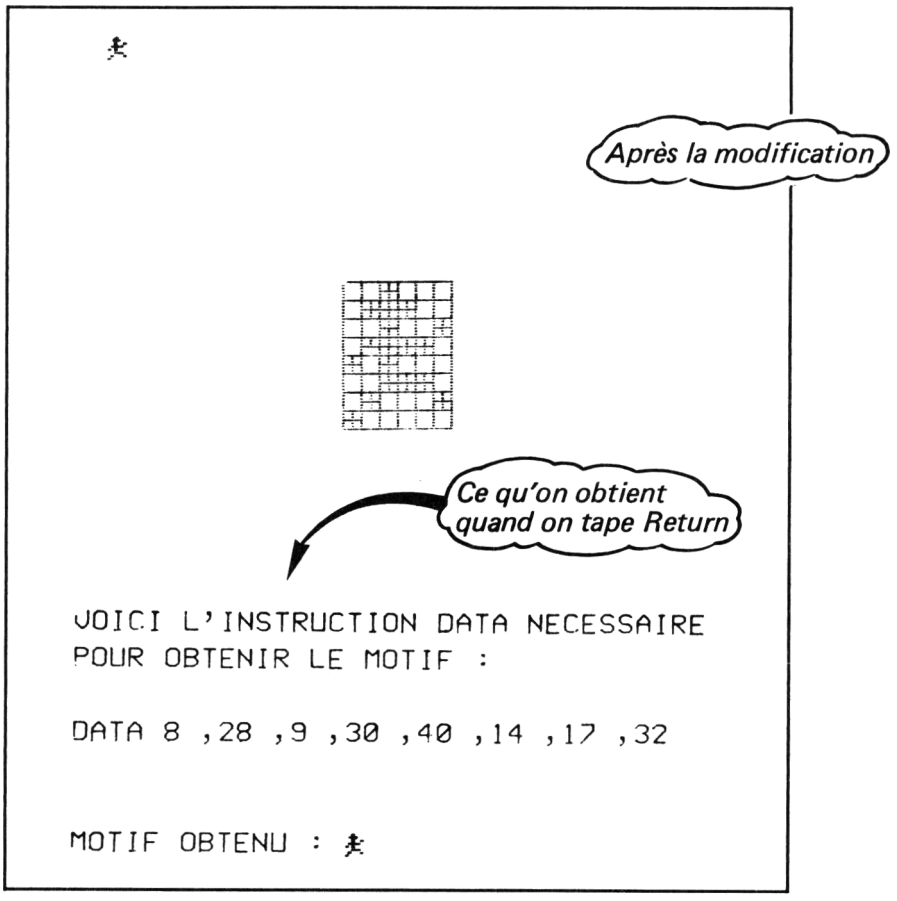
Exemple :

Pour créer un coureur à pied à la place du motif ⊏, il faut modifier le caractère 64 du jeu n° 0 :



Le crochet apparaît alors en haut à gauche de l'écran, puis une grille se remplit de croix simulant les points de ce crochet. Une croix clignote dans la case du coin supérieur gauche de la grille.

Vous pouvez déplacer cette croix dans les 4 directions afin de noircir les cases voulues (en tapant 1) ou de les blanchir (touche zéro). Pour terminer, tapez la touche **RETURN.**



Le modèle réduit est mieux que celui de la grille !

A partir de maintenant, la touche [correspond au coureur, et pour le recréer dans un programme après extinction de l'Oric, il suffit d'insérer ces cinq lignes.

```
50000 DATA 8,28,9,30,40,14,17,32
50010 FOR L=1 TO 8
50020 READ V
50030 POKE 46080+91*8+L-1,V
50040 NEXT L
```

Et voici cinq possibilités pour faire apparaître le coureur :

— taper [, mais ça n'est valable qu'en mode immédiat.
— taper :

```
PRINT "["

PLOT 10,15,91

CURSET 25,18,0
CHAR 91,0,1
```

En mode TEXT et LORES Ø

En mode haute résolution

23

COURBE DIGITALISÉE

HIRES CURSET DRAW

Pour représenter, à l'aide d'une courbe, un relevé d'observations, par exemple la taille d'un enfant selon son âge, il faut une préparation afin d'utiliser raisonnablement la feuille de papier, puis vient le report des points aux emplacements convenables ! Nous allons confier au programme suivant la préparation du travail, et il suffira de taper les coordonnées de tous les points pour qu'ils apparaissent enfin sur l'écran, à leur place respective.

```
10 CLS:PRINT
20 PRINT"TRACE D'UNE COURBE DIGITALISE
E"
30 INPUT"NOMBRE DE POINTS";NP
40 DIM X(NP),Y(NP)
50 :
60 FOR I=1 TO NP
70 PRINT "POINT NO ";I;
80 INPUT"X , Y";X(I),Y(I)
90 NEXT I
100 :
110 PRINT:PRINT"RECHERCHE DES EXTREMES
"
120 GOSUB 240
130 :
140 HIRES
150 FX=239/(XS-XI)
160 FY=199/(YS-YI)
170 FOR I=1 TO NP
180 XE=(X(I)-XI)*FX
190 YE=(YS-Y(I))*FY
200 CURSET XE,YE,1
210 NEXT I
220 :
230 END
240 :
250 REM----------
```

REMARQUES

40 — réservation pour les fichiers x et y des coordonnées des points

60 à 90 — boucle d'introduction de ces coordonnées

120 — réalisation du sous-programme de recherche des extrêmes

140 — on passe au mode haute résolution, blanc sur noir

150, 160 — facteurs d'échelle horizontaux et verticaux

170 à 210 — boucle de tracé

180, 190 — position horizontale et verticale sur l'écran

200 — allumage du point

```
260 XI=X(1)
270 XS=X(1)
280 YI=Y(1)
290 YS=Y(1)
300 :
310 FOR I=2 TO NP
320 IF X(I)<XI THEN XI=X(I)
330 IF X(I)>XS THEN XS=X(I)
340 IF Y(I)<YI THEN YI=Y(I)
350 IF Y(I)>YS THEN YS=Y(I)
360 NEXT I
370 :
380 IF XI=XS OR YI=YS THEN PRINT"REVOI
R LES COORDONNEES!":END
390 RETURN
```

REMARQUES

260 à 390 — sous-programme de recherche des extrêmes

260 à 290 — amorce de la recherche : la première valeur x et y

310 à 360 — boucle d'analyse des valeurs suivantes

380 — si on trouve une valeur plus petite que celle qu'on a retenue pour l'instant, c'est celle qu'on prend comme étant la plus petite

330 à 350 — idem pour les trois autres extrêmes

380 — si la courbe est une ligne horizontale ou verticale, on ne peut pas la cadrer dans l'écran !

Exemple :

Représentation graphique de la taille d'un individu en fonction de son âge.

Age (x) années	0	1	2	3	4	5	6	7	8
Taille (y) cm	50	72	85	92	100	106	114	118	124

Age (x) années	9	10	12	14	16	18	20	25	
Taille (y) cm	130	135	145	160	164	165	166	166	

Nombre de points : 17

```
TRACE D'UNE COURBE DIGITALISEE
NOMBRE DE POINTS? 17
POINT NO 1 X , Y? 0,50
POINT NO 2 X , Y? 1,72
POINT NO 3 X , Y? 2,85
POINT NO 4 X , Y? 3,92
POINT NO 5 X , Y? 4,100
POINT NO 6 X , Y? 5,106
POINT NO 7 X , Y? 6,114
POINT NO 8 X , Y? 7,118
POINT NO 9 X , Y? 8,124
POINT NO 10 X , Y? 9,130
POINT NO 11 X , Y? 10,135
```

```
POINT NO 12 X , Y? 12,145
POINT NO 13 X , Y? 14,160
POINT NO 14 X , Y? 16,164
POINT NO 15 X , Y? 18,165
POINT NO 16 X , Y? 20,166
POINT NO 17 X , Y? 25,166

RECHERCHE DES EXTREMES
```

AUTRE POSSIBILITÉ DE TRACÉ

On utilise pour le tracé précédent la résolution graphique maximum de l'Oric. Si l'on désire des croix au lieu de points, il suffit de modifier ainsi :

```
140 HIRES
150 FX=237/(XS-XI)
160 FY=197/(YS-YI)
170 FOR I=1 TO NP
180 XE=(X(I)-XI)*FX+1
190 YE=(YS-Y(I))*FY+1
200 CURSET XE-1,YE,1:DRAW 2,0,1
205 CURSET XE,YE-1,1:DRAW 0,2,1
210 NEXT I
```

REMARQUES

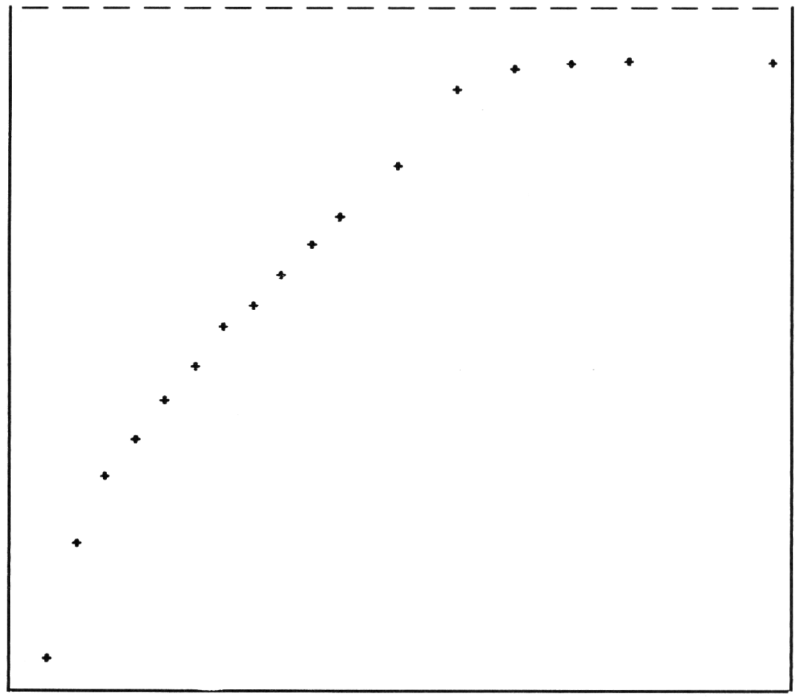
150, 160 — facteur d'échelle horizontal et vertical en s'interdisant de placer les centres de croix sur les bords de l'écran

180, 190 — position horizontale et verticale sur l'écran après décalage de 1 pour éviter le bord gauche et le haut de l'écran (il faut pouvoir les tracer ces croix !)

200 — la branche horizontale de la croix...

205 — et la branche verticale

Avec les mêmes données que précédemment on obtient :



Comment tracer une croix en forme de X au lieu d'un plus (+) ? C'est facile, il suffit de remplacer les lignes 200 et 205 par :

```
200 CURSET XE-1, YE-1, 1 : DRAW 2,2,1
205 CURSET XE-1, YE+1, 1 : DRAW 2,-2,1.
```

Nous avons entré les données dans l'ordre croissant, mais puique le programme détermine les extrêmes, vous pouvez entrer les vôtres dans n'importe quel ordre, à condition d'alterner la position horizontale et la position verticale.

24

COURBE PARAMÉTRÉE

HIRES CURSET DRAW PATTERN DEF FN

Il est parfois plus facile de représenter une courbe en reliant l'évolution des coordonnées horizontale et verticale à celle d'un paramètre auxiliaire. Le cercle en est un exemple célèbre : on obtiendra des points régulièrement espacés si l'on fait varier régulièrement la position angulaire de ceux-ci. Il faut alors définir deux expressions : celle de l'abscisse (portée ici horizontalement) et celle de l'ordonnée (position verticale). Le programme aura également comme tâche d'utiliser au mieux l'écran pour le tracé.

```
10 CLS:PRINT
20 PRINT"TRACE D'UNE COURBE PARAMETREE
"
30 DEF FNX(T)=COS(T)
40 DEF FNY(T)=SIN(T)
50 INPUT"VALEUR MIN. DU PARAMETRE";TI
60 INPUT"VALEUR MAX. DU PARAMETRE";TF
70 INPUT"NOMBRE DE POINTS";NP
80 DIM X(NP),Y(NP)
90 :
100 PRINT:PRINT"CALCUL DES COORDONNEES
"
110 T=TI
120 TPAS=(TF-TI)/(NP-1)
130 FOR I=1 TO NP
140 X(I)=FNX(T)
150 Y(I)=FNY(T)
160 T=T+TPAS
170 NEXT I
180 :
190 PRINT:PRINT"RECHERCHE DES EXTREMES
"
200 GOSUB 340
210 :
220 HIRES
230 FX=239/(XS-XI)
240 FY=199/(YS-YI)
250 FOR I=1 TO NP
260 XE=(X(I)-XI)*FX
270 YE=(YS-Y(I))*FY
280 CURSET XE,YE,1
290 NEXT I
300 :
310 END
320 :
```

REMARQUES

10 — définition de l'expression de calcul de la position horizontale d'un point (abscisse)

30 — définition de l'expression de calcul de la position verticale d'un point (ordonnée)

50, 60 — introduction des limites de variation du paramètre

70 — introduction de la finesse du tracé (nombre de points)

80 — réservation de mémoires pour les fichiers des coordonnées

20 — calcul de l'écart entre deux valeurs successives du paramètre

130 à 170 — boucle de calcul des coordonnées des points

200 — utilisation du sous-programme de recherche des extrêmes parmi les valeurs des abscisses d'une part et celles des ordonnées d'autre part

230, 240 — calcul des facteurs d'échelle

250 à 290 — boucle de tracé des points

```
330 REM-----------------------
340 XI=X(1)
350 XS=X(1)
360 YI=Y(1)
370 YS=Y(1)
380 :
390 FOR I=1 TO NP
400 IF X(I)<XI THEN XI=X(I)
410 IF X(I)>XS THEN XS=X(I)
420 IF Y(I)<YI THEN YI=Y(I)
430 IF Y(I)>YS THEN YS=Y(I)
440 NEXT I
450 :
460 IF XI=XS OR YI=YS THEN PRINT"REVOI
R LES COORDONNEES!":STOP
470 RETURN
```

REMARQUES

340 à 430 — sous-programme de recherche des extrêmes

240 à 370 — amorce de la recherche : la première valeur

390 à 440 — boucle d'analyse des valeurs suivantes

400 — si on trouve une valeur plus petite que celle qu'on a retenue pour l'instant, c'est elle qu'on prend comme étant la plus petite

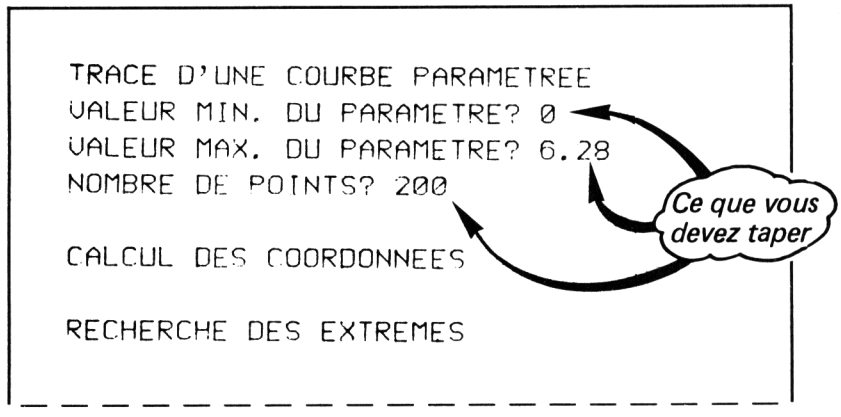
460 — puisqu'on veut utiliser au maximum l'écran, la courbe ne peut se réduire à une droite horizontale ou verticale : rejet

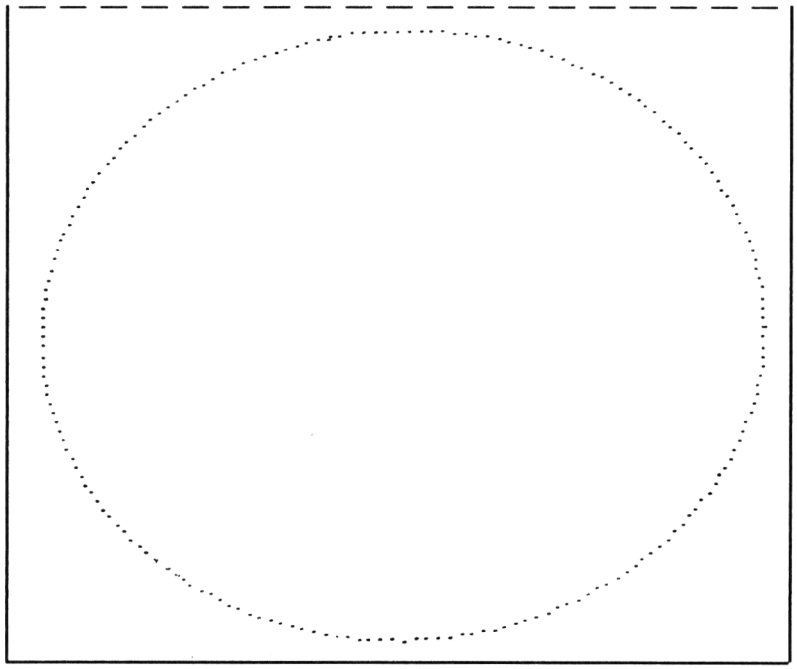
Exemple :

Pour tracer un cercle, définissons les expressions pour les positions horizontales et verticales des points

```
30 DEF FNX(T)=COS(T)
40 DEF FNY(T)=SIN(T)
```

Avec les limites zéro et 2*PI et 200 points, on obtient ce cercle aplati :





Pour comprendre pourquoi on obtient un ovale sur l'écran plutôt qu'un cercle, il faut se rappeler que notre but était d'occuper au mieux tout l'écran, et puisque celui-ci est plus large que haut... Le remède à ceci consiste à se priver d'une bande verticale de l'écran afin de travailler dans un cadre carré en modifiant le facteur d'échelle horizontal déterminé à la ligne 230. Si, de plus, on souhaite centrer la figure, il faut modifier le calcul de la position horizontale des points à la ligne 260 :

```
220 HIRES
230 FX=180/(XS-XI)
240 FY=199/(YS-YI)
250 FOR I=1 TO NP
260 XE=(X(I)-XI)*FX+30
270 YE=(YS-Y(I))*FY
280 CURSET XE,YE,1
290 NEXT I
```

REMARQUES

230 — le facteur d'échelle horizontal sera identique au facteur d'échelle vertical, puisque, pour un cercle, xmax-xmin est égal à ymax-ymin

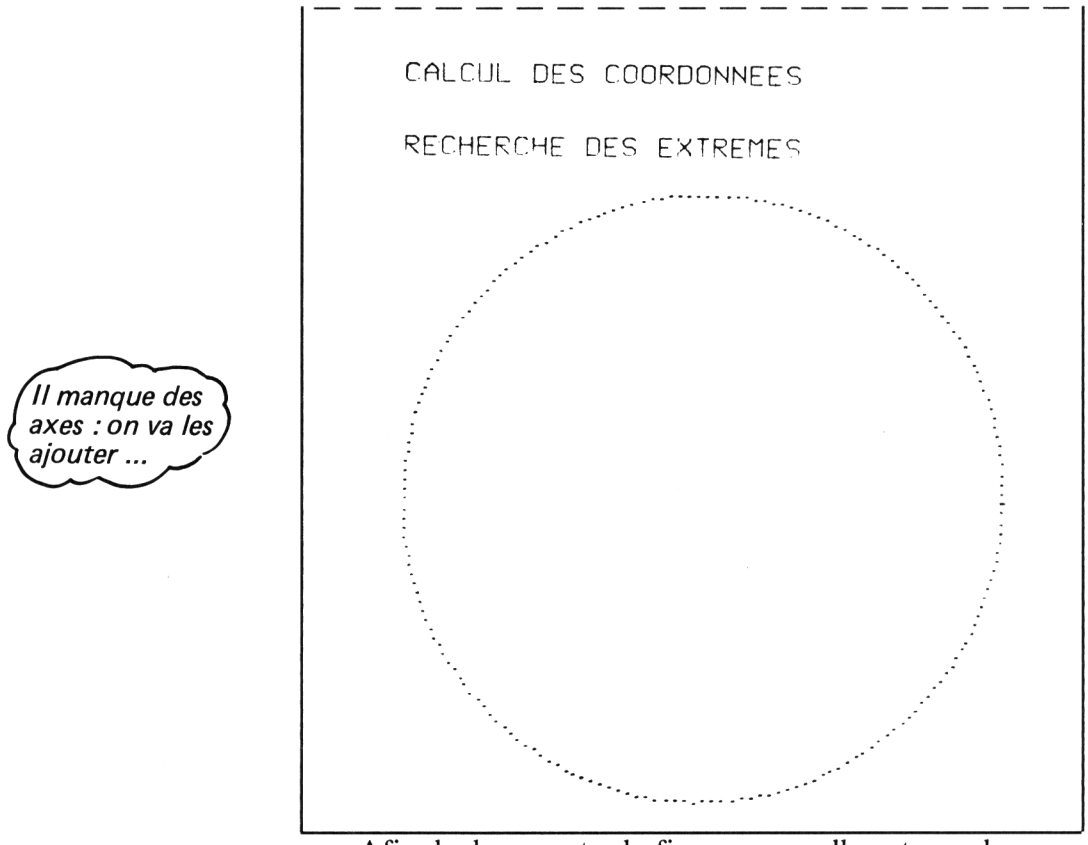
260 — position du centre décallée vers la droite

Avec les mêmes expressions :

```
30 DEF FNX(T)=COS(T)
40 DEF FNY(T)=SIN(T)
```

voici le résultat :

```
TRACE D'UNE COURBE PARAMETREE
VALEUR MIN. DU PARAMETRE? 0
VALEUR MAX. DU PARAMETRE? 6.28
NOMBRE DE POINTS? 200
```



Afin de documenter la figure, nous allons tracer les axes de coordonnées se coupant au point 0,0 en écrivant un sous-programme de tracé des axes qu'on utilisera à la ligne 300.

```
220 HIRES
230 FX=239/(XS-XI)
240 FY=199/(YS-YI)
250 FOR I=1 TO NP
260 XE=(X(I)-XI)*FX
270 YE=(YS-Y(I))*FY
280 CURSET XE,YE,1
290 NEXT I
300 GOSUB 510:REM AXES
310 :
```

REMARQUE

300 — utilisation du sous-programme de tracé des axes

```
490 :
500 REM----------------------
510 PATTERN 215
520 X0=-XI*FX
530 Y0=YS*FY
540 IF X0>=0 AND X0<=239 THEN CURSET X
0,0,1:DRAW 0,199,1
550 IF Y0>=0 AND Y0<=199 THEN CURSET 0
,Y0,1:DRAW 239,0,1
560 PATTERN 255
570 RETURN
```

REMARQUES

500 à 570 — sous-programme de tracé des axes

510 — motif du tracé

520, 530 — position sur l'écran de l'intersection des axes

540, 550 — tracé des axes horizontal et vertical, si cela est possible

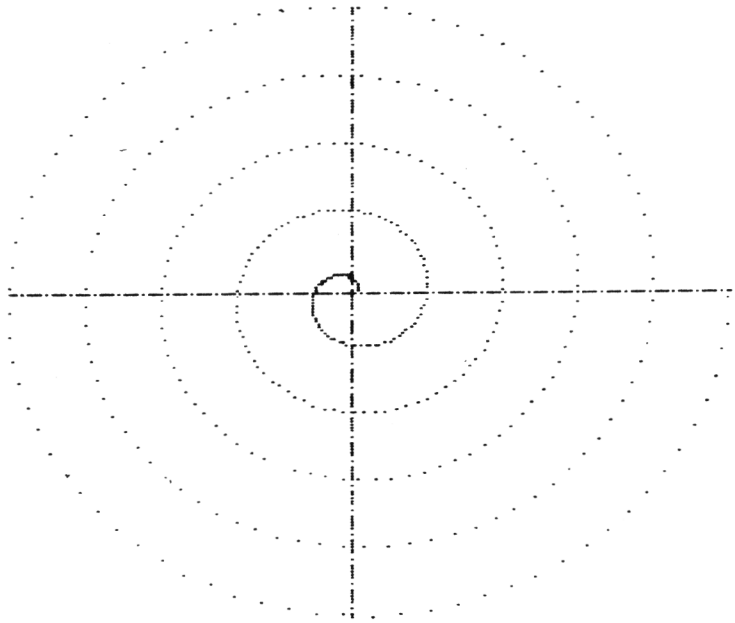
560 — on reprend le motif trait plein

Exemples :

Bien des figures peuvent être tracées telle que celle-ci avec :

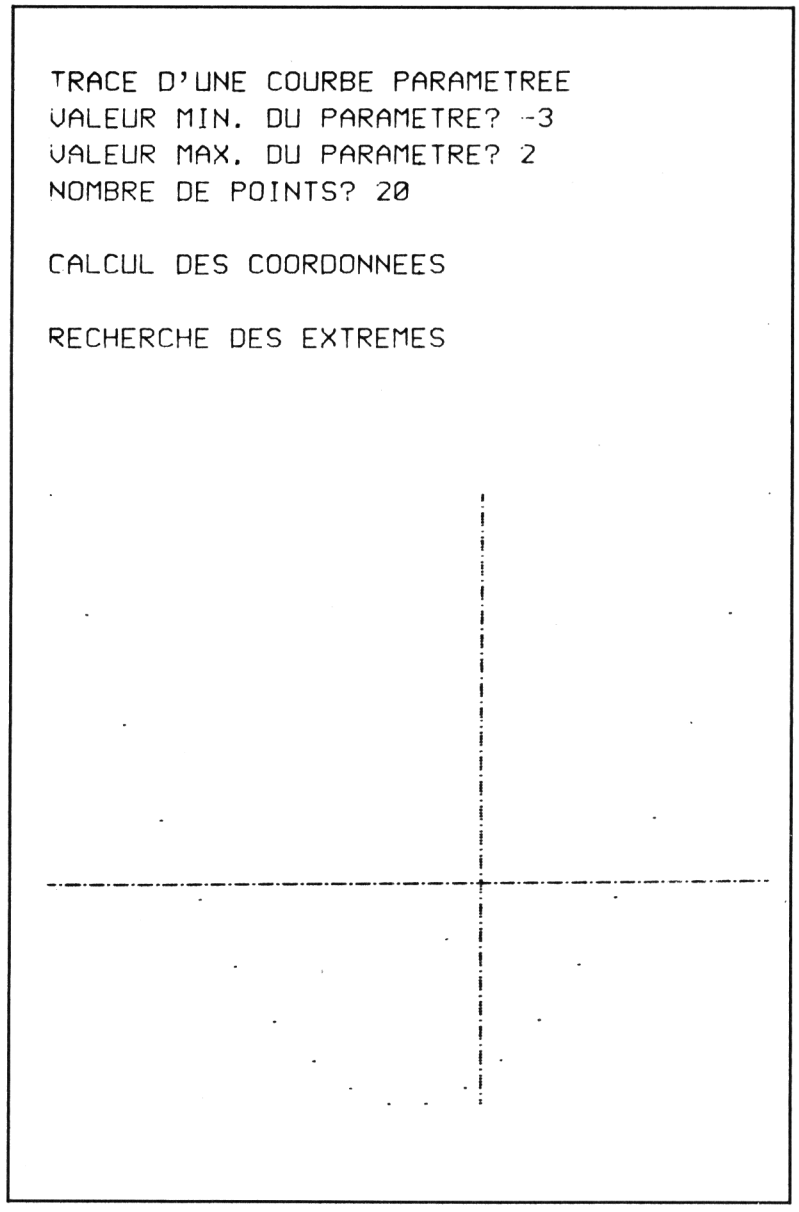
```
30 DEF FNX(T)=T*COS(T)
40 DEF FNY(T)=T*SIN(T)
```

```
TRACE D'UNE COURBE PARAMETREE
VALEUR MIN. DU PARAMETRE? 0
VALEUR MAX. DU PARAMETRE? 31,4
NOMBRE DE POINTS? 400

CALCUL DES COORDONNEES

RECHERCHE DES EXTREMES
```



ou en prenant :

```
30 DEF FNX(T)=T
40 DEF FNY(T)=(T-1)*(T+2)
```



courbe tracée entre −3 et 2 avec 20 points

25

DIAGRAMME CIRCULAIRE

HIRES CURSET CIRCLE DRAW CHAR

Pour comparer visuellement l'importance relative de plusieurs valeurs, on peut utiliser un diagramme circulaire encore appelé « camembert » ! Chaque valeur est représentée par un secteur, une part du camembert, dont la taille lui est proportionnelle. La réunion de toutes ces portions forme un disque complet.

```
10 CLS:PRINT:PRINT"DIAGRAMME CIRCULAIR
E"
20 PRINT:INPUT"NOMBRE DE VALEURS";NP
30 DIM V(NP)
40 ST=0
50 :
60 FOR I=1 TO NP
70 PRINT"VALEUR NO ";I;
80 INPUT V(I)
90 ST=ST+V(I)
100 NEXT I
110 :
120 HIRES
130 X0=114:Y0=100
140 R=60
150 CURSET X0,Y0,0
160 CIRCLE R,1
170 AE=2*PI/ST
180 DRAW R,0,1
190 :
200 SP=0:A1=0
210 FORI=1 TO NP
220 :
230 SP=SP+V(I)
240 A2=SP*AE
250 CURSET X0,Y0,0
260 DRAW R*COS(A2),-R*SIN(A2),1
270 :
```

REMARQUES

30 — réservation de mémoire pour les nb valeurs

40 — valeur initiale de S : somme des valeurs

60 à 100 — boucle d'introduction et de cumul des valeurs

90 — cumul

130 — emplacement du centre du cercle

140 — rayon du cercle

150 — le centre

160 — tracé du cercle

170 — facteur d'échelle angulaire

180 — tracé du bord horizontal de la première portion

200 — valeur initiale de SP somme progressive des valeurs, et de A1, position angulaire d'un des bords d'une portion

210 à 380 — boucle de découpage du camembert

230 — cumul des valeurs

240 — position angulaire de l'autre bord d'une portion

250, 260 — tracé de bord

```
280 AM=(A1+A2)/2
290 CURSET X0+R/2*COS(AM),Y0-R/2*SIN(A
M),0
300 IF V(I)>=0 THEN CH$=MID$(STR$(V(I)
),2) ELSE CH$=STR$(V(I))
310 FOR C=1 TO LEN(CH$)
320 CHAR ASC(MID$(CH$,C,1)),0,1
330 CURMOV 6,0,0
340 NEXT C
350 :
360 A1=A2
370 NEXT I
380 :
390 END
```

REMARQUES

280 à 340 — affichage de la valeur à l'intérieur de la portion

280 — angle moyen

290 — point de départ de l'affichage

300 — on exclut le premier caractère d'une valeur positive, car il ne plaît pas à la fonction ASC

310 à 340 — boucle d'affichage des caractères qui constituent la valeur

320 — affichage

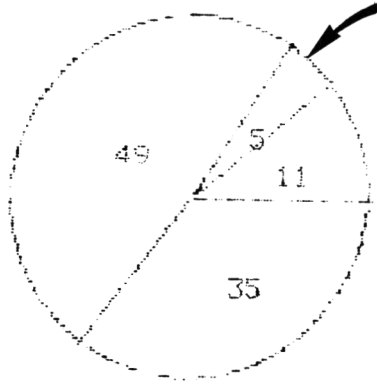
330 — on se décale pour écrire le caractère suivant

360 — le bord final de la portion devient le bord initial de la portion suivante

Exemple : diagramme circulaire de l'importance des différents moyens de transport de marchandises en 1982 :

oléoduc	11 %
voie fluviale	5 %
route	49 %
voie ferrée	35 %

```
DIAGRAMME CIRCULAIRE

NOMBRE DE VALEURS? 4
VALEUR NO 1 ? 11
VALEUR NO 2 ? 5
VALEUR NO 3 ? 49
VALEUR NO 4 ? 35
```



Faut-il regretter que la voie fluviale soit peu empruntée ?

26

FILL : DU REMPLISSAGE AVEC 64 MOTIFS

HIRES CURSET CURMOV FILL CHAR

L'instruction FILL permet de remplir rapidement une zone de l'écran soit en couleurs :

```
FILL 10,1,17
```

soit avec des motifs de hauteur réglable et que l'on peut choisir dans un jeu de 64. Chacun de ces motifs est formé de bandes verticales noires ou blanches. Pour les comparer entre eux nous allons les placer l'un en-dessous de l'autre sur quatre colonnes. En face de chacun nous écrirons son code que vous pourrez placer dans une instruction :

```
FILL 10,1,P
```

```
10 HIRES
20 PRINT:PRINT"MOTIFS OBTENUS AVEC FIL
L"
30 FOR P=32 TO 95
40 IP=INT(P/16)
50 XP=IP*30
60 YP=10*(P-IP*16)
70 CURSET XP,YP,0
80 FILL 9,1,P
90 CURSET XP+10,YP,0
100 P$=STR$(P)
110 CHAR ASC(MID$(P$,2,1)),0,1
120 CURMOV 6,0,0
130 CHAR ASC(MID$(P$,3,1)),0,1
140 NEXT P
```

REMARQUES

30 à 140 — boucle permettant d'explorer les 64 possibilités

40 — IP numérote les colonnes (0 à 3)

50, 60 — positions horizontale et verticale du coin inférieur gauche d'un motif

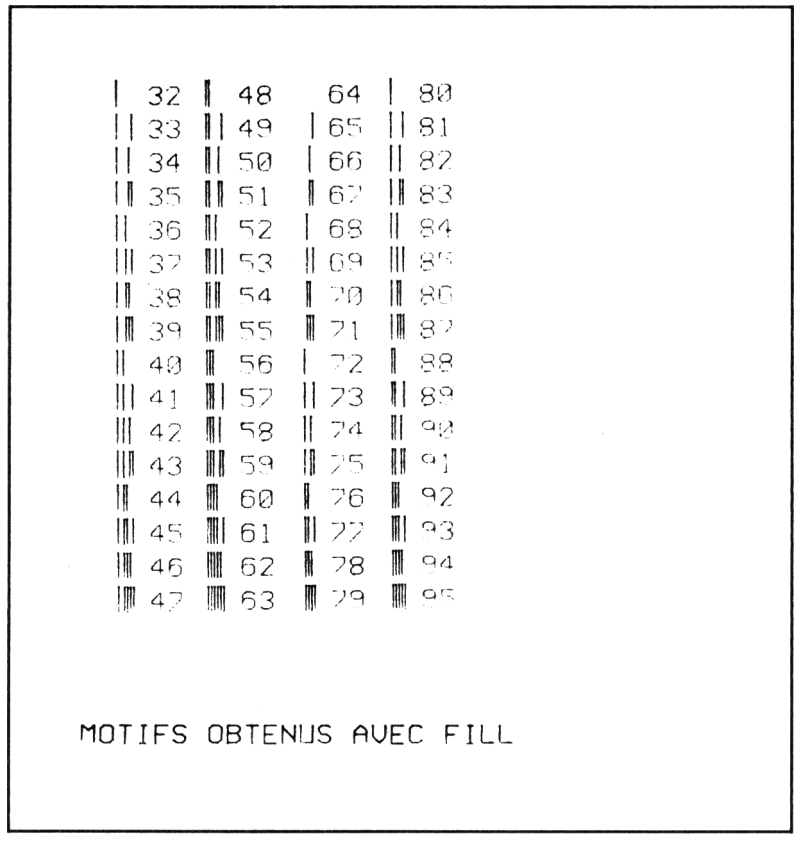
70, 80 — dessin du motif

90, 100 — préparation à l'écriture du code P

110 — le chiffre de gauche

120, 130 — celui de droite

La position des barres blanches obéit plus ou moins à un système de numération binaire. Faites votre choix :



REPONSES AUX QUESTIONS SUR LE CHAPITRE VII

1. En HIRES, la définition de l'écran est de 200 rangées sur 240 colonnes.

2. CURSET sert à positionner le curseur.

3. CURMOV déplace le curseur par rapport à sa position antérieure.

4. CIRCLE est accompagné de son rayon et du nombre FD. (Son centre est fourni par la position du curseur.)

5. DRAW tire un trait entre la position du curseur et un point dont on donne les coordonnées en valeurs relatives.

6. FILL sert non seulement à mettre de la couleur mais commande aussi le clignotement, le placement de motifs graphiques, etc.

7. REPEAT UNTIL commande la répétition d'une séquence tant que la condition posée n'est pas réalisée.

8. Si N vaut 7, la condition *N supérieure à 8* n'est pas remplie et, par conséquent, le branchement en 70 ne se fait pas. Donc, le programme passe en 60 et affiche BONJOUR.

9. L'ordre KEY$ n'arrête pas le programme : lorsque le Basic le rencontre, à quelqu'instant que ce soit, il lit le clavier.

CHAPITRE VIII

LE MOMENT MUSICAL

Nous vous avons déjà montré comment programmer les quatre « bruits » qui existent en instructions. Maintenant, c'est de la « musique » que nous allons faire. On examinera aussi, dans ce chapitre, quelques dernières instructions en Basic qui vous permettront de vous lancer sans hésiter dans la programmation.

Principaux thèmes abordés

SOUND	Enveloppe des sons
MUSIC	Accord musical
PLAY	Période avec SOUND
Notes musicales	GET
Canaux des sons	

1. MUSIC, pour commencer

SOUND
MUSIQUE
PLAY

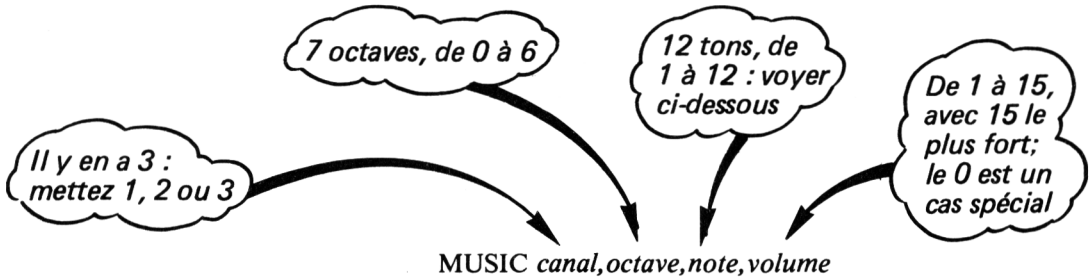
Pour faire de la musique, vous disposez de trois commandes qui sont SOUND (pour *son*), MUSIC et PLAY (pour *jouer*). En première approche :

• SOUND sert à définir un son. Cet ordre est accompagné de 3 paramètres. Si le dernier n'est pas nul, le son est émis pour cet ordre, mais s'il est égal à zéro, le son est « mis en mémoire » et exécuté par un ordre PLAY suivant.

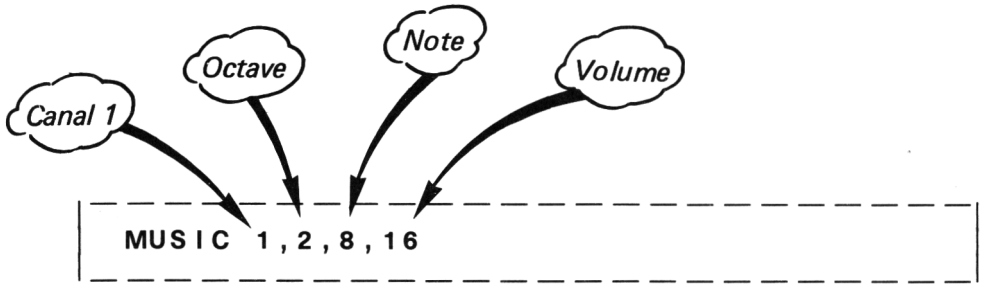
• MUSIC, lui, ne définit pas n'importe quel son mais les *notes de musique de la gamme*. Son *dernier* paramètre joue comme avec SOUND.
• PLAY exécute des sons (SOUND ou MUSIC) définis antérieurement.

MUSIC

Commençons par l'ordre MUSIC. Pour jouer des notes de la gamme, il faut définir la note, bien sûr, mais indiquer aussi dans quelle octave elle se situe, quel canal de l'Oric elle emploiera et enfin, quelle sera sa puissance sonore (son volume). Ainsi, MUSIC doit être accompagné de 4 paramètres dans cet ordre :



Notez les virgules de séparation : elles sont obligatoires. Commencez par essayer, en mode direct :

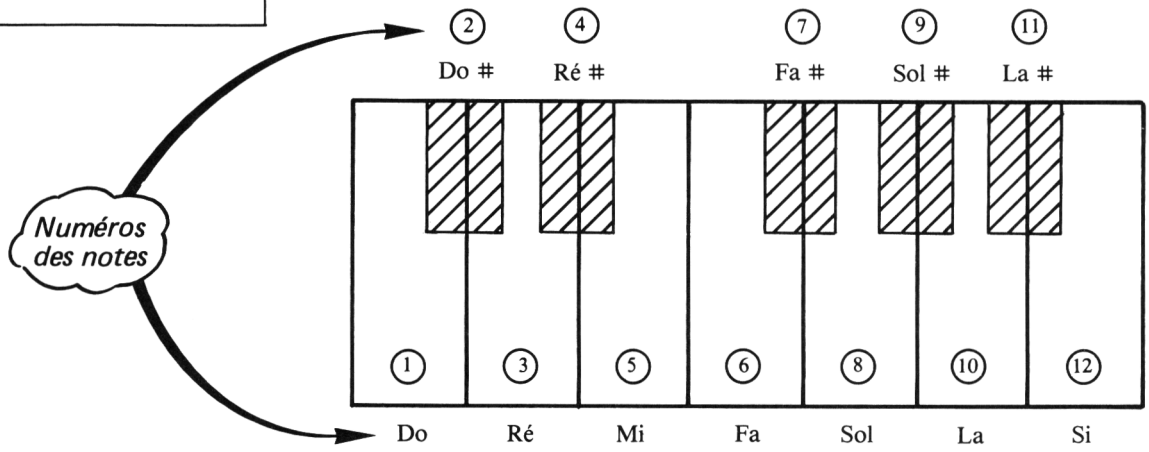


Au RETURN vous reconnaîtrez un sol, 2e octave (si vous êtes musicien). Pour stopper le son, frappez sur n'importe quelle touche, la barre d'espacement de préférence. Si les touches ne sont pas sonorisées, tapez d'abord CTRL/F.

Les octaves sont numérotées de Ø à 6, le Ø étant le plus grave.

NOTES MUSICALES

Les notes, elles, sont données ainsi, si l'on reprend l'exemple d'un clavier de piano :



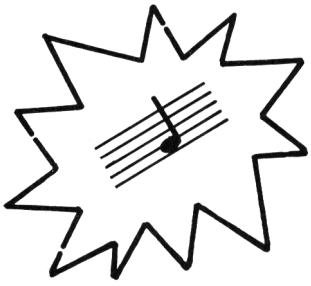
Le dernier paramètre de MUSIC situe le volume sonore. Essayez tout d'abord quelques combinaisons avec l'ordre MUSIC, en mode direct mais avec le canal 1.

On va voir que, pour exploiter les 3 canaux sonores de l'Oric, il faut employer l'instruction PLAY. Donc, pour l'instant, restez-en au canal 1.

En fait, le plus simple consiste encore à passer par le court programme suivant, qui va vous permettre de modifier à chaque fois tous ces paramètres :

```
10 REM *MUSIC*
20 INPUT "OCTAVE=";O
30 INPUT "NOTE=";N
40 INPUT "VOLUME=";V
50 MUSIC 1,O,N,V
60 GOTO 20
```

Les lignes 2Ø à 4Ø vont, grâce à l'ordre INPUT qui combine implicitement un PRINT de ce qui est contenu entre guillemets, vous demander ces divers paramètres ; voici ce que donnerait un RUN :



```
RUN

OCTAVE=?  3
NOTE=?  5
VOLUME=?  12
OCTAVE=?
```

Faites RETURN après chaque entrée

Au dernier RETURN, la note programmée retentit, mais aussi ...

... La question suivante apparait : dès que vous frapperez une touche, la note émise cessera

Expérimentez plusieurs combinaisons, afin de bien maîtriser les valeurs des paramètres : pour terminer, faites CTRL/C.

2. Pour jouer : PLAY

PLAY

Si le dernier paramètre de MUSIC est un Ø, l'ordre MUSIC n'est plus exécuté et dépend d'un autre ordre, PLAY. Celui-ci, à son tour, est accompagné de paramètres au nombre de 4 :

1) — *Le canal,* qui détermine le, ou les canaux actifs (de Ø à 7).

2) — *Le « bruit »,* qui agit sur ces sons bizarres qu'on appelle bruits, programmés avec SOUND et non avec MUSIC (1 à 7).

3) — *L'enveloppe,* qui agit comme *bruit* mais cette fois, sur les sons.

4) — *La durée :* c'est le temps pour l'exécution de « l'enveloppe » ou du « bruit ».

Reprenons en détaillant, cette fois.
• Les canaux sont ainsi numérotés :

CANAUX DES SONS

Programmation des canaux avec PLAY

Numéro	Canaux actifs
Ø	Aucun
1	1
2	2
3	1 et 2
4	3
5	1 et 3
6	2 et 3
7	1, 2 et 3

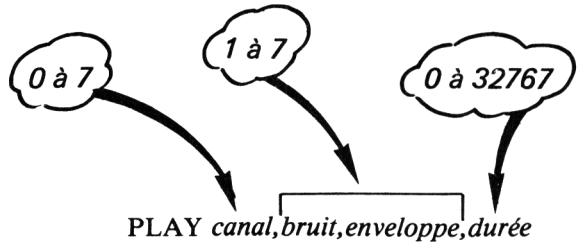
• Le « bruit » ou « l'enveloppe » définissent ce qu'en termes techniques, on appelle la *courbe enveloppe* du son et qu'on pourrait ainsi interpréter :

ENVELOPPE DES SONS

Enveloppe des sons et des bruits

Numéro	Action
1 2	Son décroissant (durée déterminée) Son croissant (durée déterminée)
3	Son périodique, continu, en « dents de scie »
4	Son périodique, continu, « ondulé »
5	Son nom périodique, continu, difficile à définir
6	Son périodique, continu, en dents de scie inverses de 3
7	Son nom périodique, continu, différent de 5 et aussi difficile à définir. Mais on va écouter tout cela.

• La durée, enfin, va de Ø à 32 767 ; pour comprendre son rôle, on va l'expérimenter.
La syntaxe de PLAY est ainsi :



A nouveau, toutes ces explications écrites sont indigestes si on ne pratique pas. Aussi, pour les expérimenter, va-t-on vous proposer un programme du même type que le précédent, ou le PLAY sera combiné à un MUSIC 1,2,8,Ø (notez que le dernier paramètre est un zéro, afin de *passer la main* au PLAY qui suit). Procédons cependant ici paramètre par paramètre pour commencer ; nous avons volontairement séparé ces divers programmes d'essai afin que vous puissiez y revenir ensuite. Voyons pour la *durée :*

```
200 REM --DUREE DE Ø A 32767--
210 MUSIC 1,2,8,Ø
220 INPUT "DUREE";D
230 PLAY 1,Ø,1,D
240 GOTO 2ØØ

RUN 2ØØ
```

Le 0 «passe la main» à PLAY pour exécution

N'oubliez pas de préciser le RUN 200

Pour PLAY, on a choisi le canal 1 et l'enveloppe 1

Au RUN, le programme vous demande la durée. Commencez par frapper 15ØØ : vous obtiendrez un *sol* comparable à celui d'un piano (le son s'éteint très vite). Moins de 5ØØ, le son est trop bref. Plus de 5ØØØ, il n'en finit plus de s'atténuer et de mourir. Essayez plusieurs valeurs, puis faites CTRL/C.
Expérimentons, maintenant, l'*enveloppe* avec une durée de 15ØØ. Voici le nouveau programme :

```
300 REM -ENVELOPPE, DE 1 A 7-
310 MUSIC 1,2,8,0
320 INPUT "ENVELOPPE";E
330 PLAY 1,0,E,1500
340 GOTO 310

RUN 300
```

Avec enveloppe = 1, vous retrouvez la frappe d'un piano. Avec 2, c'est l'inverse : le son démarre faible et s'amplifie jusqu'à son maximum, cela pendant la « durée » de 1500 (une durée supérieure lui accorderait davantage de temps).
Avec 3, c'est un marteau qui frappe continuellement la note sol ; comme si vous faisiez 1 en permanence. Le 4, lui, vous donne à peu près le 2 en répétition continue. Le 5 démarre bizarrement, un peu comme en 1, puis reste sur un son continu. Le 6 ressemble au 3, mais est encore moins harmonieux peut-être. Enfin, le 7 fournit une montée rapide en puissance, comme le 2, et se poursuit par un son continu. Ces explications sont approximatives : rien ne vaut un essai.
Franchissons encore un pas, en travaillant à la fois sur *l'enveloppe* et la *durée* avec la séquence suivante :

```
100 REM -ENVELOPPE ET DUREE-
110 MUSIC 1,2,8,0
120 INPUT "ENVELOPPE";E
130 INPUT "DUREE";D
140 PLAY 1,0,E,D
150 GOTO 120

RUN 100
```

Essayez plusieurs combinaisons ; pour l'enveloppe 1 et 2, le résultat est évident : le « piano » ou un son croissant, durant ce que vous fixez avec D. Essayez avec :

• *Enveloppe 3 :* pour durée = 1000, c'est une enclume sur laquelle on frappe à coups répétés. A 10000, c'est la cloche d'une église (l'auteur fait un peu de poésie). A 100, c'est un moteur qui ne tourne pas rond. Le son se répète à chaque fois à perpétuité, jusqu'à une nouvelle frappe.

• *Enveloppe 4 :* avec D = 1000, c'est un avion qui marche très mal (ou une scierie) ; avec D = 10000, on obtient un son qui commence très fort, diminue, disparaît, puis réaugmente, atteint son maximum, puis rediminue, etc. Si D = 100, c'est un teuf-teuf rapide.

• *Enveloppe 5 :* avec D = 1000, c'est un « point » suivi d'un son continu ; avec 100, on ne l'entend plus, mais le son est toujours continu ensuite.

• *Enveloppe 6 :* reportez-vous au 3, à peu près.

• *Enveloppe 7 :* comme 5, mais au lieu d'un « point », c'est un son croissant.

En conclusion, la durée, avec l'enveloppe 5 et 7, agit sur la courte période préparatoire du son, mais plus sur la suite.
Si, de plus, vous faites des essais avec des « enveloppes » non prévues, vous risquez d'obtenir toutes sortes de choses curieuses et indescriptibles.

3. Des accords

ACCORD MUSICAL

Il est temps, maintenant, de vous montrer comment actionner plusieurs canaux simultanément. Pour cela, il faudra employer obligatoirement PLAY. Par exemple, pour travailler avec les 3 canaux, on sera en PLAY 7 pour le premier paramètre. Un accord *do, mi, sol* se programmerait ainsi :

```
NEW

5 REM *ACCORD*
10 MUSIC 1,2,1,10
20 MUSIC 2,2,5,10
30 MUSIC 3,2,8,10
40 PLAY 7,0,0,0

RUN
```

DO
MI
Sol

Cette fois, on a programmé la puissance avec MUSIC ; c'est PLAY qui met en service les trois canaux (faute de quoi, seul le canal 1 fonctionnerait).

Ici, PLAY ne sert qu'à programmer les canaux ; c'est pourquoi ses autres paramètres sont à zéro. D'autres valeurs ne joueraient pas sur le résultat.

Essayez de remplacer le 7 de PLAY par d'autres valeurs ; à chaque fois, vous entendrez les canaux mis en service, 2 ou 1. Par contre, le 0 les interdit tous ; on va s'en servir pour programmer cet accord sur 1 seconde, puis pour l'arrêter ; ajoutez 50 et 60 :

```
5 REM *ACCORD*
10 MUSIC 1,2,1,10
20 MUSIC 2,2,5,10
30 MUSIC 3,2,8,10
40 PLAY 7,0,0,0
50 WAIT 100
60 PLAY 0,0,0,0
```

Pendant une seconde ...
... Puis retour au silence

Vous pouvez, maintenant, programmer « Au clair de la lune » ou des mélodies plus « savantes » si vous le désirez.

4. Sons divers

SOUND

Il nous reste à examiner l'ordre SOUND, qui agit comme MUSIC. On doit préciser : *le canal, la période* et *le volume :*

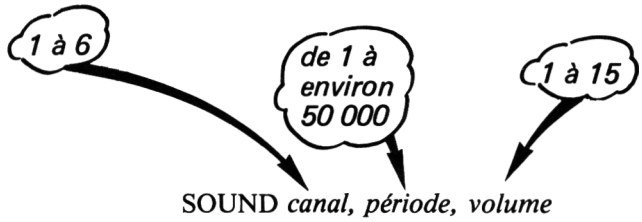
• *Le canal* sera, cette fois :
— 1, ou 2, ou 3 pour des sons purs
— 4, 5 ou 6 pour des « bruits ».
En fait, il n'y a qu'un canal pour les bruits, le *canal 4.* Aussi 4, 5 et 6 reviennent à spécifier quel canal son (1, 2 ou 3) doit être mêlé au canal 4.

PERIODE AVEC SOUND

• *La période* est un terme technique : elle fixe la hauteur du son. Plus sa valeur est forte et plus le son est grave. Chaque note de la gamme est déterminée par une valeur précise de la *période ;* avec SOUND, vous pourrez imaginer toutes les notes possibles.

• *Enfin, le volume* va de 1 à 15, comme avec MUSIC. A zéro, il transmet l'exécution à PLAY.

La syntaxe de SOUND est ainsi :



Commençons par expérimenter la période avec ce programme :

```
NEW

10 REM -PERIODE DE SOUND-
20 INPUT "PERIODE";P
30 SOUND 1,P,10
40 GOTO 20
```

Canal — Volume

Au RUN, entrez plusieurs valeurs ; vous constaterez que les hauteurs de sons courants se situent autour des centaines. Terminez par CTRL/C.
Pour mettre les divers canaux en service, il faut, à nouveau, passer par PLAY. Mais alors là, les combinaisons sont quasi-infinies et nous ne pouvons vous les décrire. Voici, par exemple, comment obtenir un coup de feu (sans SHOOT) :

```
NEW

10 SOUND 5,500,10
20 PLAY 7,1,0,500
```

Chaque nouveau coup de feu vous imposera un RUN. Voici comment « tirer » au moment que vous choisirez.

5. Attendez une touche

GET

Il existe une méthode servant à attendre la frappe d'une touche pour déclencher des opérations. On passe alors par l'ordre GET ; lorsque le Basic rencontre GET, il stoppe toute exécution et attend la frappe d'une touche. Vous pouvez sélectionner une touche particulière, ou aucune en particulier, et dans ce cas, toutes seront prises en compte. Dès que vous en frappez une, le Basic poursuivra alors son programme. Voici ce que cela donnerait avec notre coup de feu :

On n'a défini aucune touche en particulier. Dès la frappe d'une d'elles, le programme se poursuit

```
5 REM --COMMANDE DE FEU--
8 GET T$
10 SOUND 5,500,10
20 PLAY 7,1,0,500
30 GOTO 8
```

La valeur correspondant à la touche frappée s'appellera T$

Après GET, on donne généralement un nom de chaîne à la touche frappée. Vous pourriez, ensuite, vérifier par programme que T$ correspond à telle touche que vous avez sélectionnée pour le tir, par exemple la touche ↑ (et pourquoi pas).

Dès le RUN, le programme attend ici la frappe d'une touche. Dès qu'elle se produit, le coup de feu part. Hélas : le bruit propre des touches, qui sont sonorisées, perturbe notre coup de feu. Il faudrait supprimer cette sonorisation avec un CTRL/F avant le RUN, comme on l'a vu (on la rétablira avec CTRL/F aussi).
Mais essayez donc, maintenant, de stopper ce programme : toutes les touches ou combinaisons de touches, y compris CTRL/C, déclenchent un coup de feu ! Avez-vous une idée sur la façon de vous en sortir ? Oui, c'est cela : retournez l'Oric et appuyez sur le RESET. Vous pouvez alors faire CTRL/F, puis RUN, à nouveau. Constatez qu'en maintenant n'importe quelle touche enfoncée, vous simulerez une mitrailleuse.
A partir de cela, vous pourriez imaginer le programme suivant : vous passez en mode graphique et vous déplacez un mobile coloré sur l'écran. Il n'est vulnérable que sur certaines positions que vous déterminez. Si vous tirez à ce moment, vous le désintégrez ! Et si ce projet ne vous convient pas, rien ne vous interdit d'en concevoir d'autres !
Toujours est-il que vous en connaissez maintenant assez pour passer aux programmes que nous vous proposons à la fin de chaque chapitre, et à ceux qui sont regroupés dans le chapitre 9. Vous pourrez alors broder sur ces programmes et innover en écrivant les vôtres.

Pour la musique trois instructions c'est peu et c'est beaucoup à la fois ! SOUND, MUSIC et PLAY ont en effet bien des possibilités, mais il faut du temps et de la patience pour les découvrir. Si on veut comparer l'effet produit par SOUND 1, 2, 3 par exemple, il faut taper une bonne vingtaine de touches, de quoi perdre le fil !. Pour évaluer les richesses de SOUND, voici un programme qui ne vous demandera que ce que vous voulez changer à chaque nouvel essai. Nous ferons la même chose pour MUSIC et PLAY pour qu'il n'y ait pas de jaloux !

QUESTIONS SUR LE CHAPITRE VIII

Il est évidemment peu intéressant de répondre à des questions sur les sons sans manipuler... Voici cependant ce que vous devez conserver à l'esprit :

1. *SOUND et MUSIC peuvent-il provoquer l'émission de sons, sans recours à PLAY ?*
   ☐ *Oui*
   ☐ *Non*

2. *MUSIC sert à créer :*
   ☐ *Des sons de timbre absolument quelconque*
   ☐ *Uniquement les notes de gammes musicales*

3. *Quel ordre provoque la mise en service de plusieurs canaux simultanément ?*
   ☐ *MUSIC*
   ☐ *PLAY*

4. *Pour créer des bruits divers, on utilisera :*
   ☐ *SOUND*
   ☐ *MUSIC*

5. *L'ordre GET :*
   ☐ *Sert à lire le clavier, sans arrêter le programme, à la volée*
   ☐ *Stoppe le programme et attend qu'une touche soit frappée.*

Réponses page 194

27

SOUND : DÉCOUVREZ LES SONS
SOUND GET

SOUND a besoin de trois paramètres pour produire un son : le numéro du canal, la période et le volume. Au démarrage du programme qui suit c'est le canal 1 qui produira un son musical de période 1000 avec le volume 3. Ensuite vous changerez à volonté un de ces trois paramètres en conservant les deux autres. Pour chaque son produit le programme affichera l'instruction SOUND complète, comme si vous l'aviez tapé au clavier. Notez les combinaisons qui vous plaisent, vous les placerez dans vos programmes... Un son durera jusqu'à ce qu'on modifie un paramètre. Pour terminer taper CTRL C.

```
10 C=1:P=1000:V=3
20 REPEAT
30 CLS
40 SOUND C,P,V
50 PRINT"SOUND ";C;",";P;",";V
60 GET T$
70 IF T$="C" THEN INPUT"CANAL 1 A 6";C

80 IF T$="P" THEN INPUT"PERIODE 0 A 32
767";P
90 IF T$="V" THEN INPUT"VOLUME 0 A 15"
;V
100 UNTIL T$=CHR$(3)
```

REMARQUES

10 — les paramètres de départ

20 à 100 répéter jusqu'à ce qu'on tape CTRL C

30 — le ménage sur l'écran

40 — le son

50 — l'image

60 — attente d'une frappe de touche

70 à 90 — les trois choix possibles

Au départ, vous obtenez :

```
SOUND 1 ,1000 ,3
```

Pour changer le canal, tapez C puis le numéro.
Pour la période c'est P et le volume V :

```
SOUND 1 ,200 ,6
```

Essayez plusieurs canaux, et vous remarquerez qu'il n'y a guère que le 1 qui donne quelque chose, pour l'instant. Que se passe-t-il si on multiplie ou si on divise la période par 2 ?

Testez votre oreille : diminuez la période en gardant le volume 3.
Si vous arrivez à une période de 5, c'est très bien, entre 6 et 8 ça va ; au-dessus, c'est peut-être le haut-parleur qui faiblit ! Pourquoi ne pas raccorder l'Oric à votre chaîne hi-fi ou un appareil de ce genre ?...

28

MUSIC : MUSIQUE
MUSIC GET

MUSIC a besoin de 4 paramètres : numéro de canal, octave, note et volume.

Adoptons le même principe que pour le programme 27 : un choix entre 4 frappes de touches pour préciser le paramètre qu'on veut changer, C, O, N ou V. Puis on tape la valeur de ce paramètre (suivie de return) et le son retentit accompagné du texte de l'instruction MUSIC qu'il aurait fallu taper.

```
10 C=1:O=2:N=1:V=3
20 REPEAT
30 CLS
40 MUSIC C,O,N,V
50 PRINT"MUSIC ";C;",";O;",";N;",";V
60 GET T$
70 IF T$="C" THEN INPUT"CANAL 1 A 6";C

80 IF T$="O" THEN INPUT"OCTAVE 0 A 7";
O
90 IF T$="N" THEN INPUT"NOTE 1 A 12";N

100 IF T$="V" THEN INPUT"VOLUME 0 A 15
";V
110 UNTIL T$=CHR$(3)
```

REMARQUES

10 — les paramètres de départ

20 à 110 — répéter jusqu'à ce qu'on tape CTRL C

40 — le son

50 — l'image

60 — attente d'une frappe de touche

70 à 100 — étude des 4 cas

Pour commencer :

Ce qu'on obtient après RUN →

```
MUSIC 1, 2, 1, 3
```

Montez la gamme : fixez l'octave et passez en revue les notes 1 à 12, puis recommencez pour l'octave au-dessus.

En tapant, par exemple, la lettre O, puis 3 suivi de RETURN : →

```
MUSIC 1 ,3 ,1 ,3
```

Ici encore, seul le canal 1 fonctionne.

Nous découvrirons les autres grâce au concours de l'instruction PLAY... Mais pourriez-vous couvrir 8 octaves comme le fait déjà MUSIC ?

29

SOUND MUSIC PLAY : POUR MELANGER LES SONS

SOUND MUSIC PLAY

Sans l'instruction PLAY, SOUND et MUSIC ne délivrent qu'un seul son.

Pour apprécier les nombreuses possibilités de l'ensemble PLAY, SOUND, MUSIC, nous vous proposons un programme dans la lignée des deux précédents, mais beaucoup plus long, car le domaine est vaste !

Pour chacune des trois instructions, vous pourrez choisir de modifier un des paramètres. Afin d'y voir plus clair, on affichera dans la moitié inférieure de l'écran (rouge), l'instruction SOUND ou MUSIC la plus récente concernant chacun des 6 canaux, et on mettra à jour le texte de l'instruction PLAY.

Le haut de l'écran (vert) servira de journal de bord relatant les diverses directives que vous donnerez.

```
10 PLAY 0,0,0,0
20 POKE 623,27
30 PAPER 1:CLS
40 POKE 623,14
50 POKE 619,18
60 CLS
70 DIM P(6),O(6),N(6),V(6)
80 B$="                        "
90 FOR C=1 TO 6
100 N(C)=1
110 NEXT C
120 :
130 REPEAT
140 PRINT:PRINT"S:SOUND, M:MUSIC, P:PLAY
, F:FIN?"
150 GET T$
160 IF T$="S" THEN GOSUB 1000:T$=""
170 IF T$="M" THEN GOSUB 2000:T$=""
180 IF T$="P" THEN GOSUB 3000:T$=""
190 UNTIL T$="F"
200 END
210 :
```

REMARQUES

10 — arrêtez la musique !

20 — plein écran

30 — fond rouge, partout

40 — écran réduit : les 14 lignes du haut

50 — fond vert pour le haut

70 — réservation pour les paramètres période, octave, note, volume

80 — pour blanchir plus tard : 23 espaces

90 à 110 — initialisation des notes à 1 pour les six canaux

130 à 190 — répétition jusqu'à ce qu'on tape F pour arrêter

140 — le premier choix

150 — attente d'une frappe de touche

160 — SOUND

170 — MUSIC

180 — PLAY

```
1000 PRINT"SOUND"
1010 PRINT"C:CANAL, P:PERIODE, V:VOLUME?
"
1020 GET T$
1030 IF T$="C" THEN INPUT"CANAL 1 A 6";C
:GOTO 1010
1040 IF T$="P" THEN INPUT"PERIODE 0 A 32
767";P(C)
1050 IF T$="V" THEN INPUT"VOLUME 0 A 15"
;V(C)
1060 PLOT 1,13+2*C,B$
1070 A$="SOUND "+STR$(C)+","+STR$(P(C))+
","+STR$(V(C))
1080 PLOT 1,13+2*C,A$
1090 SOUND C,P(C),V(C)
1100 RETURN
1990 :
2000 PRINT"MUSIC"
2010 PRINT"C:CANAL, O:OCTAVE, N:NOTE, V:
VOLUME?"
2020 GET T$
2030 IF T$="C" THEN INPUT"CANAL 1 A 3";C
:GOTO 2010
2040 IF T$="O" THEN INPUT"OCTAVE 0 A 7";
O(C)
2050 IF T$="N" THEN INPUT"NOTE 1 A 12";N
(C)
2060 IF T$="V" THEN INPUT"VOLUME 0 A 15"
;V(C)
2070 PLOT 1,13+2*C,B$
2080 A$="MUSIC "+STR$(C)+","+STR$(O(C)
)+","+STR$(N(C))+","+STR$(V(C))
2090 PLOT 1,13+2*C,A$
2100 MUSIC C,O(C),N(C),V(C)
2110 RETURN
2990 :
3000 PRINT"PLAY"
3010 PRINT"N:CANAL NOTE, B:CANAL BRUIT
, F:FORME, D:DUREE?"
3020 GET T$
3030 IF T$="N" THEN INPUT"CANAL NOTE 0
 A 7";CN:GOTO 3010
3040 IF T$="B" THEN INPUT"CANAL BRUIT
0 A 7";CB:GOTO 3010
3050 IF T$="F" THEN INPUT"FORME 1 A 7"
;F
3060 IF T$="D" THEN INPUT"DUREE 0 A 32
767";D
```

REMARQUES

1000 à 1100 — sous-programme d'utilisation de l'instruction SOUND

1010 — sous-choix

1020 — attente d'une frappe de touche

1030 — choix du canal

1040 — choix de la période

1050 — choix du volume

1060 — effaçage de l'ancienne instruction SOUND

1070 — texte de la nouvelle

1080 — affichage

1090 — le son

REMARQUES

2000 à 2110 — sous-programme d'utilisation de l'instruction MUSIC

2010 — sous-choix

2020 — attente d'une frappe de touche

2030 — choix du canal

2040 — choix de l'octave

2050 — choix de la note

2060 — choix du volume

2070 — effaçage de l'ancienne instruction MUSIC

2080 — texte de la nouvelle

2090 — affichage

2100 — le son

REMARQUES

3000 à 3210 — sous-programme d'utilisation de l'instruction PLAY

3010 — sous-choix

3020 — attente d'une frappe de touche

3030 — choix du ou des canaux de sons purs

3040 — idem pour les canaux bruits

3050 — forme

3060 — durée

```
3110 FOR I=1 TO 3
3120 NC=CN AND 2^(I-1)
3130 BC=CB AND 2^(I-1)
3140 IF NC THEN PLOT 0,13+2*I,"*" ELSE
PLOT 0,13+2*I," "
3150 IF BC THEN PLOT 0,19+2*I,"*" ELSE
PLOT 0,19+2*I," "
3160 NEXT I
3170 PLOT 15,20,B$
3180 A$="PLAY "+STR$(CN)+","+STR$(CB)+
","+STR$(F)+","+STR$(D)
3190 PLOT 15,20,A$
3200 PLAY CN,CB,F,D
3210 RETURN
```

REMARQUES

3110 à 3160 — boucle de marquage des canaux concernés par PLAY

3120 — NC vaut 1 pour un canal son concernés par PLAY

3130 — idem pour un canal bruit

3140 — repérage par une étoile si le canal son n° 1 est concerné

3150 — idem pour le canal bruit n° I+3

3170 — effaçage de l'ancienne instruction PLAY

3180 — texte de la nouvelle

3190 — affichage

3200 — exécution

Et maintenant vous allez être récompensé de votre effort, puisque vous pourrez essayer confortablement toutes les combinaisons qui vous passent par la tête, à condition qu'elles soient possibles. Pour ne pas allonger le programme, nous n'avons pris aucune sécurité : si vous demandez l'impossible, vous obtiendrez un message d'erreur et le programme s'interrompra. Ce n'est pas grave, tapez GOTO 130, et tout repartira comme avant !

Exemple :

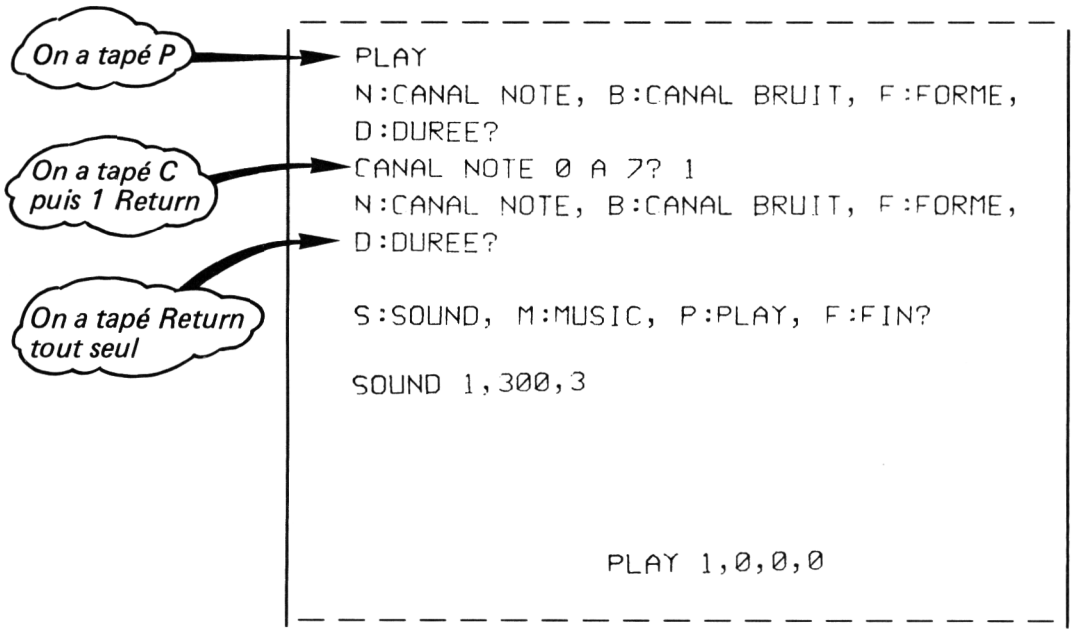
Tapez RUN puis

S C 1 Return P 300 Return

```
S:SOUND, M:MUSIC, P:PLAY, F:FIN?
SOUND
C:CANAL, P:PERIODE, V:VOLUME?
CANAL 1 A 6? 1
C:CANAL, P:PERIODE, V:VOLUME?
PERIODE 0 A 32767? 300

S:SOUND, M:MUSIC, P:PLAY, F:FIN?



SOUND 1,300,0
```

Il manque le volume et le lancement par PLAY ; tapez S V 3 Return (on ne reprécise pas le canal, puisque c'est le même) puis P, vous devez avoir ceci sur l'écran :

```
VOLUME 0 A 15? 3

S:SOUND, M:MUSIC, P:PLAY, F:FIN?
```



Voici le résultat du mélange des 3 canaux de son pur avec du bruit sur le canal 3, en travaillant sur ce canal avec la forme répétitive 3 et une durée de 500 ;

```
S:SOUND, M:MUSIC, P:PLAY, F:FIN?
PLAY
N:CANAL NOTE, B:CANAL BRUIT, F:FORME,
D:DUREE?
DUREE 0 A 32767? 500

S:SOUND, M:MUSIC, P:PLAY, F:FIN?


*MUSIC 1,4,1,5

*MUSIC 2,4,5,5

*MUSIC 3,4,8,0
                PLAY 7,4,3,500



*SOUND 6,100,0
```

Mais surtout, ne vous arrêtez pas là : tâtonnez.

Quand c'est bon notez la recette, et lorsque c'est franchement mauvais tapez F puis RUN pour obtenir le silence !

30

EN RYTHME

Un bruit bien cadencé peut marquer le rythme. Mélangez avec une note bien choisie et vous obtenez de la musique !

Le choix du rythme se fera en tapant un chiffre entre 1 et 7, quant aux notes on les trouvera sur la deuxième rangée de touches à partir du haut. Si vous les comptez, de Q à], vous trouverez 12, c'est-à-dire exactement le nombre de demi-tons d'une octave. Le choix de celle-ci vous est laissé au début.

```
10 CLS;PRINT"EN RYTHME !"
20 DATA Q,W,E,R,T,Y,U,I,O,P,[,]
30 DIM N(93)
40 FOR C=65 TO 93
50 N(C)=1
60 NEXT C
70 FORI=1 TO 12
80 READ C$
90 N(ASC(C$))=I
100 NEXT I
105 :
110 PRINT:PRINT"TEMPO : 1 A 7"
120 PRINT:PRINT"NOTE Q A ]"
130 INPUT"QUELLE OCTAVE (0 A 7)";O
140 SOUND 5,100,0
145 :
150 REPEAT
160 T$=KEY$
170 IF T$>"0" AND T$<"8" THEN PLAY 1,2
,3,100*(VAL(T$)+2)
180 IF T$>="A" AND T$<="]" THEN MUSIC
1,O,N(ASC(T$)),8
190 UNTIL T$=CHR$(3)
```

REMARQUES

20 — les 12 touches produisant une note

30 — N fichier des notes

40 à 60 — on met 1 pour chaque note, a priori...

70 à 100 — ... et on change les 12 prévues

110, 120 — affichage des possibilités

130 — choix de l'octave

140 — préparation du bruit

150 à 190 — répétition jusqu'à ce qu'on tape CTRL C

160 — lecture du clavier au vol

170 — choix d'un rythme pour le canal 1, la durée est de 300, pour 2 : 400 etc...

180 — choix d'une note : une des 12 prévues, ou bien 1

Exemple :

Choisissez un rythme de 1 (rapide) à 7 (lent) puis pianotez en utilisant les touches Q, W, ... P, [et].

```
EN RYTHME !

TEMPO : 1 A 7

NOTE Q A ]
QUELLE OCTAVE (0 A 7)? 4
```

Ne vous gênez pas pour changer le volume dans l'instruction MUSIC (ligne 180) ou la période de SOUND (ligne 140) !

31

MUSIQUE AU HASARD
MUSIC

Reposons-nous un instant en laissant le hasard créer une musiquette en trois mouvements : léger et aigu, moins aigu, puis plus posé et plus grave. Pour obtenir ce résultat, il faudra bien sûr forcer un peu la main au hasard ! Pour voir les différents mouvements, les notes seront affichées sur l'écran à la bonne hauteur.

On laissera le choix entre deux octaves, en fixant la plus basse. La durée de chaque note sera la même pour les deux premiers mouvements, et le double pour le troisième.

```
10 DATA 2,3,10
20 DATA 2,2,10
30 DATA 2,1,20
40 L=30
50 CLS
60 PLOT 1,26,"MOUVEMENT NUMERO "
70 X=1
80 :
90 FOR M=1 TO 3
100 PLOT 18,26,STR$(M)
110 READ PLAGE,DECAL,DM
120 FOR K=1 TO L/DM
130 O=INT(RND(1)*PLAGE)+DECAL
140 FOR I=1 TO 5
150 D=RND(1)*DM+1
160 N=INT(RND(1)*12)+1
170 H=27-(O-1)*6-N/2
180 PLOT X,H,"♀"
190 MUSIC 1,O,N,8
200 WAIT D
210 X=X+1
220 NEXT I
230 NEXT K
240 WAIT 50
250 NEXT M
260 :
270 PLAY 0,0,0,0
```

REMARQUES

10 à 30 — un guide pour le hasard

40 — réglage de la longueur de chaque mouvement

70 — position horizontale des notes : 4 pour la première

90 à 250 — boucle sur les mouvements

100 — affichage du numéro du mouvement

110 — lecture des valeur guides : plage et décalage pour l'octave, durée maximum d'une note

120 à 230 — boucle pour ajuster la longueur d'un mouvement

130 — choix de l'octave pour cinq notes successives

140 à 220 — boucle sur cinq notes

150 — durée d'une note

160 — valeur de la note

170 — sa position verticale sur l'écran

180 — on la "note"

190 — on la joue

200 — on la maintient

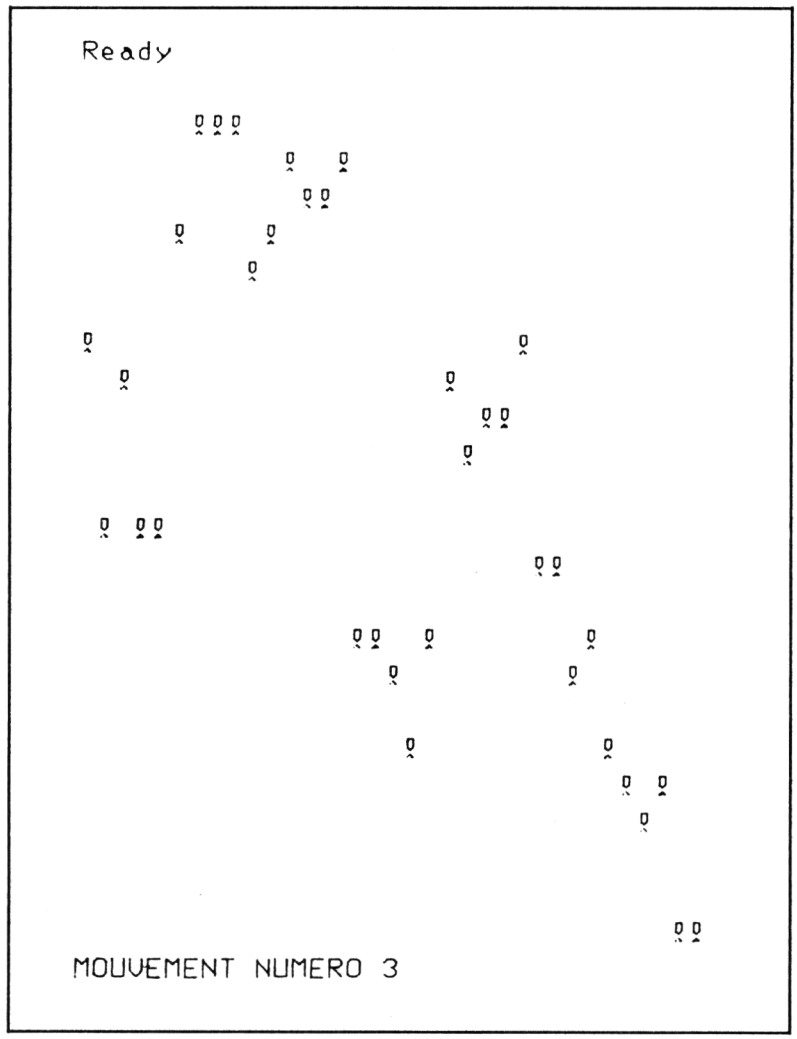
210 — position horizontale de la prochaine note

240 — séparation entre deux mouvements

270 — fin : silence !

Exemple :

Avec les trois instructions DATA données dans le texte du programme, voici ce que nous avons obtenu. La "photo" prise avant la fin signale que le dernier mouvement était le troisième, et on a droit à un magnifique Ready en haut de l'écran puisque le programme n'utilise pas d'instruction PRINT.



Modifiez les valeurs des DATA, en retouchant éventuellement à la ligne 170 si les notes ne tiennent plus sur l'écran.

REPONSES AUX QUESTIONS SUR LE CHAPITRE VIII

1. Oui, à la condition que leur dernier paramètre, le volume, ne soit pas nul.

2. MUSIC sert à créer des notes musicales, et ce, sur plusieurs gammes.

3. C'est PLAY qui sert à mettre en service plusieurs canaux simultanément, et non MUSIC.

4. Pour créer des bruits, il faut utiliser SOUND..

5. GET stoppe le programme et attend qu'une touche soit frappée. Ne confondez pas GET avec KEY$ qui, lui, fait lire le clavier au passage, sans arrêter le programme.

CHAPITRE IX

POUR REVISER CINQ PROGRAMMES UTILES OU DIVERTISSANTS

A la fin des chapitres 4 à 8, vous avez trouvé plusieurs exemples de programmes qui mettaient en pratique les nouveautés dévoilées dans le chapitre, sans faire appel aux notions traitées par la suite.

C'est maintenant qu'il faut regrouper tout ce que vous savez en prenant comme support cinq programmes dans lesquels vous devriez retrouver des têtes connues ! Chaque fois, après avoir fixé l'objectif, et pour vous aider à concevoir par la suite vos propres programmes, nous analyserons le travail en dégageant les grandes lignes. Et, comme précédemment, le texte du programme sera accompagné de remarques, et nous vous donnerons des exemples d'utilisation !

32

MISE EN FORME : COMMENT AFFICHER PROPREMENT UN NOMBRE

Vous vous êtes peut-être posé cette question : y a-t-il un moyen pour afficher un nombre représentant une somme d'argent en francs et centimes, au lieu d'avoir quatre ou cinq chiffres après la virgule comme ça peut arriver. On souhaite alors écrire (sur l'écran) selon un format imposé.

Dans certains dialectes du Basic, ou dans d'autres langages, on peut trouver une instruction qui satisfait à ce besoin légitime. Mais le Basic de l'Oric ne connaît pas cette instruction, et il n'est pas le seul !

Il faut donc faire le travail, et on distingue deux méthodes : celle du tout numérique et celle du traitement de texte. La première méthode consiste à arrondir la valeur en imposant le nombre de décimales : pour garder les 2 premiers chiffres après la virgule ; ce qui doit donner 3.52 dans 3,523 on peut écrire

```
INT(3.523*100)/100
```

Mais au moment de diviser pour retrouver le bon ordre de grandeur, on a parfois des surprises : pour écrire 1 avec 5 décimales, INT(1*10∧5)/10∧5 donne 0.999999999 à cause de l'erreur sur 10∧5 ! Et même lorsque ça se passe bien, on ne maîtrise pas la place qu'occupe le résultat : elle dépend du nombre de chiffres à gauche de la virgule.

L'autre méthode est plus sûre et plus complète : elle consiste à se servir des fonctions de traitements de texte du Basic (LEFT$, MID$, RIGHT$, LEN). Elle est aussi plus lourde.

Pour étudier les caractères qui composent un nombre, il faut transformer celui-ci en chaîne de caractères avec la fonction STR$. Après avoir trouvé la place du point décimal, on peut maîtriser entièrement l'affichage des valeurs numériques) : nombre de décimales et nombre de caractères maximum utilisés pour représenter les valeurs, y compris les espaces éventuels. Le programme qui suit est constitué schématiquement de deux parties :

— mise en forme (lignes 9000 à 9210)
— utilisation de cette mise en forme à titre d'essai (lignes 10 à 140).

Il existe au moins deux cas pour lesquels ça ne marche pas : pour un nombre avec puissance de 10 (E..) et lorsque la dernière décimale conservée vaut 9 et que la suivante est supérieure à 4. Nous réglerons ce deuxième inconvénient dans la seconde version du programme.

```
10 CLS:PRINT:PRINT"AJUSTAGE D'UN NOMBRE"
20 INPUT"VOTRE NOMBRE ";N
30 GOSUB 9000
40 REPEAT
50 PRINT:INPUT"NB DE CARACTERES ";NC
60 INPUT"NB DE DECIMALES  ";ND
70 GOSUB 9050
80 :
90 PRINT"RESULTAT ";N$
100 INPUT"AUTRE MISE EN FORME (O/N)";R$
110 UNTIL R$="N"
120 :
130 END
140 :


899  REM-----------------------------
9000 B$="          "
9010 E$="**********"
9020 Z$="0000000000"
9030 RETURN
9040 :


9045 REM-----------------------------
9050 N$=STR$(N):L=LEN(N$)
9060 IF N>=0 THEN L=L-1:N$=RIGHT$(N$,L)
9070 :
9080 FOR C=1 TO L
9090 IF MID$(N$,C,1)="." THEN R=C:GOTO 9
140
9100 NEXT C
9110 N$=N$+"."
```

REMARQUES

20 — introduction de la valeur à mettre en forme

30 — utilisation du sous-programme d'initialisation

40 à 100 — répétition jusqu'à l'abandon

50 à 60 — introduction des caractéristiques de mise en forme

70 — utilisation du sous-programme de mise en forme

90 — affichage du résultat obtenu

100 — question

130 — fin

9000 à 9030 — sous-programme d'initialisation

9000 — chaîne servant de réservoir de blancs (9 espaces)

9010 — réservoir d'étoiles

9020 — réservoir de zéros

9050 à 9210 — sous-programme de mise en forme N→N$

9050 — la valeur numérique N est convertie en chaîne de caractères sous le nom N$, L : nombre de caractères

9060 — nombre positif : on rejette le caractère de gauche

9080 à 9100 — boucle de recherche du point

9090 — R représente la place du point, s'il existe, comptée à partir de la gauche

9110 — le point n'existe pas : on l'ajoute à droite

```
9120 L=L+1:R=L
9130 :
9140 IF ND=0 THEN N$=LEFT$(N$,R-1):GOTO
9190
9150 N$=N$+Z$
9160 DE$=MID$(N$,R+ND,1)
9170 IF MID$(N$,R+ND+1)>="5" THEN DE$=CH
R$(ASC(DE$)+1)
9180 N$=LEFT$(N$,R+ND-1)+DE$
9190 L=LEN(N$)
9200 IF L>NC THEN N$=LEFT$(E$,NC) ELSE N
$=LEFT$(B$,NC-L)+N$
9210 RETURN
```

REMARQUES

9120 — il y a donc un caractère de plus et R marque le dernier

9140 — on ne souhaite pas de décimales : il ne faut conserver que ce qui est à gauche du point

9150 — on souhaite des décimales : on ajoute neuf zéros au bout du nombre pour satisfaire tous les besoins légaux (<=9 décimales)

9160, 9170 — arrondissement de la dernière décimale à conserver, en tenant compte de la suivante : si celle-ci est supérieure ou égale à 5, on ajoute 1 à la dernière décimale conservée

9180 — troncature : on ne conserve que ND décimales

9200 — le nombre de caractères nécessaires pour afficher la valeur est supérieur au nombre prévu (NC) : afficher NC étoiles, sinon, on complète à gauche par des espaces, si c'est nécessaire, afin de former une chaîne de NC caractères

Exemples :

```
AJUSTAGE D'UN NOMBRE
VOTRE NOMBRE ? 12.3456

NB DE CARACTERES ? 7
NB DE DECIMALES  ? 4
RESULTAT 12.3456
AUTRE MISE EN FORME (O/N)? O

NB DE CARACTERES ? 6
NB DE DECIMALES  ? 4
RESULTAT ******
AUTRE MISE EN FORME (O/N)? O

NB DE CARACTERES ? 4
NB DE DECIMALES  ? 1
RESULTAT 12.3
AUTRE MISE EN FORME (O/N)? O

NB DE CARACTERES ? 2
NB DE DECIMALES  ? 0
RESULTAT 12
AUTRE MISE EN FORME (O/N)? N
```

```
AJUSTAGE D'UN NOMBRE
VOTRE NOMBRE ? -12.3456

NB DE CARACTERES ? 6
NB DE DECIMALES  ? 2
RESULTAT -12.35
AUTRE MISE EN FORME (O/N)? O

NB DE CARACTERES ? 5
NB DE DECIMALES  ? 2
RESULTAT *****
AUTRE MISE EN FORME (O/N)? O

NB DE CARACTERES ? 7
NB DE DECIMALES  ? 2
RESULTAT  -12.35
AUTRE MISE EN FORME (O/N)? N
```

TABULATION AVEC CADRAGE A DROITE

Nous allons illustrer la mise en forme de nombres en regroupant dans une même table un nombre, son carré, sa racine carrée et le sinus de ce nombre (représentant un angle exprimé en radians). La première partie de ce programme (lignes 10 à 150) gère les calculs, la tabulation et fait appel à deux sous-programmes (lignes 9000 à 9240) pour la mise en forme de chaque nombre à afficher. Ce principe s'applique à n'importe quelle tabulation.

Il faut résoudre aussi le défaut signalé : 1.298 arrondi à 2 décimales devient 1.2 : puisqu'on ajoute 1 au code du 9, on obtient le code des deux points ! Une solution consiste à repasser provisoirement à la valeur numérique pour faire cet arrondi avant de finir la mise en forme, comme on l'a déjà fait.

```
10 CLS:PRINT:PRINT"TABULATION AVEC CADRA
GE A DROITE"
20 PRINT"  X     X^2    SQR(X)    SIN(X)"
30 PRINT
40 GOSUB 9000
45 :
50 FOR X=1 TO 9
60 N=X:NC=4:ND=1:GOSUB 9050
70 PRINT N$;
80 N=X^2:NC=5:ND=2:GOSUB 9050
90 PRINT"  ";N$;
100 N=SQR(X):NC=6:ND=4:GOSUB 9050
110 PRINT"  ";N$;
120 N=SIN(X):NC=6:ND=4:GOSUB 9050
130 PRINT"   ";N$
140 NEXT X
145 :
150 END
160 :
899 REM--------------------------
9000 B$="         "
9010 E$="**********"
9020 Z$="000000000"
9030 RETURN
9040 :


9045 REM-----------------------------
9050 N$=STR$(N):L=LEN(N$)
9060 IF N>=0 THEN L=L-1:N$=RIGHT$(N$,L)
9070 :
9080 FOR C=1 TO L
9090 IF MID$(N$,C,1)="." THEN R=C:GOTO 9
140
9100 NEXT C
9110 N$=N$+"."
9120 L=L+1:R=L
9130 :
```

REMARQUES

40 — utilisation du sous-programme d'initialisation

50 à 140 — boucle permettant de faire varier le nombre contenu dans X

60 — préparation des données nécessaires au sous-progrmme de mise en forme, et utilisation de celui-ci pour X

70 — affichage du résultat de cette mise en forme

80 — même travail qu'à la ligne 60, pour le carré de X

90 — affichage du carré de X, mis en forme

100, 110 — idem pour la racine carrée de X

120, 130 — idem pour le sinus de X

150 — fin

9000 à 9030 — sous-programme d'initialisation

9000 — chaîne servant de réservoir de blancs (9 espaces)

9010 — réservoir d'étoiles

9020 — réservoir de zéros

9050 à 9240 — sous-programme de mise en forme de N→N$

9050 — la valeur numérique N est convertie en chaîne de caractères sous le nom N$,L: nombre de caractères

9060 — nombre positif : on rejette le caractère de gauche

9080 à 9100 — boucle de recherche du point

9090 — R représente la place du point, s'il existe, comptée à partir de la gauche

9110 — le point n'existe pas : on l'ajoute à droite

9120 — il y a donc un caractère de plus et R marque le dernier

```
9140 IF ND=0 THEN N$=LEFT$(N$,R-1):GOTO
9190
9150 N$=N$+Z$
9160 IF MID$(N$,R+ND+1)<"5" THEN GOTO 92
10
9170 N=VAL(N$)
9180 NP=ABS(N)+10^(-ND)
9190 N$=STR$(SGN(N)*NP)
9200 IF N>0 THEN N$=MID$(N$,2)
9210 N$=LEFT$(N$,R+ND)
9220 L=LEN(N$)
9230 IF L>NC THEN N$=LEFT$(E$,NC) ELSE N
$=LEFT$(B$,NC-L)+N$
9240 RETURN
```

REMARQUES

9140 — on ne souhaite pas de décimales : il ne faut conserver que ce qui est à gauche du point

9150 — on souhaite des décimales : on ajoute neuf zéros au bout du nombre pour satisfaire tous les besoins légaux (< = neuf décimales)

9170 à 9200 — arrondissement de la dernière décimale à conserver, en tenant compte de la suivante : si celle-ci est supérieure ou égale à 5, on ajoute 1 à la dernière décimale conservée

9210 — troncature : on ne conserve que ND décimales

9230 — si le nombre de caractères nécessaires pour afficher la valeur est supérieure au nombre prévu (NC) : afficher NC étoiles, sinon complément à gauche par des espaces, si nécessaire, afin de former une chaîne de NC caractères

La table obtenue est agréable à lire, et vous pouvez la remodeler en changeant les valeurs de NC et ND :

```
TABULATION AVEC CADRAGE A DROITE
  X     X^2    SQR(X)    SIN(X)

 1.0    1.00   1.0000     .8415
 2.0    4.00   1.4142     .9093
 3.0    9.00   1.7321     .1411
 4.0   16.00   2.0000    -.7568
 5.0   25.00   2.2361    -.9589
 6.0   36.00   2.4495    -.2794
 7.0   49.00   2.6458     .6570
 8.0   64.00   2.8284     .9894
 9.0   81.00   3.0000     .4121
```

33

CLASSEMENT ALPHABÉTIQUE

La mise en mémoire d'un mot se faisant selon une codification normalisée (baptisée ASCII), la lettre A a une valeur inférieure à la lettre B, et ainsi de suite. Ceci conduit à un classement des codes correspondant à celui des lettres. De plus, la comparaison de deux chaînes de caractères, par exemple deux mots, est réalisée en Basic caractère par caractère : si le caractère de gauche est le même dans les deux chaînes, la comparaison porte sur le suivant, et ainsi de suite jusqu'à ce qu'une différence soit rencontrée. Si les deux chaînes n'ont pas la même longueur, la plus courte est inférieure à la plus longue.

Grâce à tout ceci, le classement de deux mots, ou groupes de mots (l'espace compte alors), est possible au sens alphabétique : le classement se fera dans l'ordre croissant : ''ABC'' est inférieur à ''ABC'', et ''ABC'' est inférieur à ''ABCD''. Lorsqu'il y a plus de deux mots à classer, une méthode simple à appliquer consiste à chercher d'abord le mot qui doit venir en tête dans l'ordre alphabétique : c'est le plus petit lorsqu'on le compare aux autres. On l'échange alors dans le fichier de mots avec celui qui se trouvait en tête, et on recommence sans tenir compte du premier mot, qui est maintenant bien classé. Il suffit de s'arrêter lorsque les deux derniers mots viennent d'être échangés (si nécessaire).

```
10 CLS:PRINT"CLASSEMENT D'UN FICHIER D
E MOTS"
20 INPUT"NOMBRE DE MOTS ";NM
30 DIM M$(NM)
40 FOR N=1 TO NM
50 INPUT"MOT";M$(N)
60 NEXT N
70 CLS:PRINT"FICHIER CLASSE":PRINT
80 FOR T=1 TO NM-1
90 N$=M$(T):P=T
100 FOR N=T+1 TO NM
110 IF M$(N)<N$ THEN N$=M$(N):P=N
120 NEXT N
130 M$(P)=M$(T)
140 M$(T)=N$
150 PRINT N$
160 NEXT T
170 PRINT M$(NM)
```

REMARQUES

20 — introduction du nombre de mots

30 — réservation de mémoire pour le fichier de mots

40 à 60 — boucle d'introduction des mots

80 à 160 — boucle permettant de trouver successivement les mots dans l'ordre croissant

90 — hypothèse : le Tème mot est plus petit que les suivants, P indique la position du plus petit mot

100 à 120 — boucle de recherche du plus petit mot compris entre le Tème et le dernier

110 — si un mot plus petit est trouvé, on le loge dans N$ et on change le contenu de P

130, 140 — échange

150 — affichage du mot bien classé

170 — affichage du dernier mot

Exemple :

```
CLASSEMENT D'UN FICHIER DE MOTS
NOMBRE DE MOTS ? 10
MOT? POIRE
MOT? POMME
MOT? PAMPLEMOUSSE
MOT? PECHE
MOT? PRUNE
MOT? ABRICOT
MOT? FRAISE
MOT? ORANGE
MOT? KAKI
MOT? FRAMBOISE




FICHIER CLASSE

ABRICOT
FRAISE
FRAMBOISE
KAKI
ORANGE
PAMPLEMOUSSE
PECHE
POIRE
POMME
PRUNE
```

Si vous préférez les fromages, ne vous gênez pas !

CLASSEMENT ALPHABÉTIQUE AVEC NOMBRES ASSOCIÉS

Il est fréquent qu'on associe une ou plusieurs informations à chaque mot d'une liste. Lorsqu'on souhaite classer ces mots par ordre alphabétique, il ne faut pas séparer les informations qui s'y rattachent, et pourtant celles-ci ne doivent pas être prises en compte pour le classement. La méthode employée ici consiste à utiliser un fichier pour les mots et un fichier pour chaque information associée. Tout échange nécessaire dans le fichier de mots donnera lieu à un échange identique dans chaque fichier associé. Nous prendrons le cas d'une seule information associée.

```
10 CLS:PRINT"CLASSEMENT DE DEUX FICHIE
RS"
20 INPUT"NOMBRE DE MOTS ";NM
30 DIM M$(NM),NB(NM)
40 FOR N=1 TO NM
50 INPUT"MOT";M$(N)
55 INPUT"NOMBRE";NB(N)
60 NEXT N
70 CLS:PRINT"RESULTAT DU CLASSEMENT":P
RINT
80 FOR T=1 TO NM-1
90 N$=M$(T):P=T
100 FOR N=T+1 TO NM
110 IF M$(N)<N$ THEN N$=M$(N):P=N
120 NEXT N
130 M$(P)=M$(T)
140 M$(T)=N$
145 NA=NB(T):NB(T)=NB(P):NB(P)=NA
150 PRINT NB(T),M$(T)
160 NEXT T
170 PRINT NB(NM),M$(NM)
```

REMARQUES

20 — introduction du nombre de chaînes de caractères

30 — réservation de mémoire pour le fichier des mots et des valeurs numériques

40 à 60 — boucle d'introduction des informations

80 à 160 — boucle permettant de trouver successivement les mots dans l'ordre croissant

90 — hypothèse : le Tème mot est plus petit que les suivants, P indique la position du plus petit mot

100 à 120 — boucle de recherche du plus petit mot compris entre le Tème et le dernier

110 — si un mot plus petit est trouvé, on le loge dans N$ et on change le contenu de P

130, 140 — échange de mots

145 — échange correspondant pour les valeurs numériques

150 — affichage du mot et du nombre classés

170 — affichage pour le dernier mot

Exemple :

Chaque mot du langage possède un code compris entre 128 et 255. On en trouve parfois la liste classée dans l'ordre croisant des codes, mais il serait souvent plus intéressant de l'avoir par ordre alphabétique :

```
CLASSEMENT DE DEUX FICHIERS
NOMBRE DE MOTS ? 10
MOT? END
NOMBRE? 128
MOT? EDIT
NOMBRE? 129
MOT? INVERSE
NOMBRE? 130
MOT? NORMAL
NOMBRE? 131
MOT? TRON
NOMBRE? 132
```

```
RESULTAT DU CLASSEMENT

129     EDIT
128     END
130     INVERSE
137     LORES
131     NORMAL
135     PLOT
134     POP
136     PULL
133     TROFF
132     TRON
```

Ce n'est qu'un extrait de la liste des mots et fonctions du langage

34

COURSE DE HAIES

Pour entretenir les réflexes, rien de tel qu'une course de haies dans un fauteuil : en appuyant sur une touche du clavier au bon moment, le coureur sautera par dessus la haie. Mais si vous appuyez trop tôt ou trop tard, il renversera la haie. Le parcours comporte 12 haies réparties sur quatre lignes droites.

Ces haies seront représentées par la lettre I en position debout, et par le signe – lorsqu'elles ont été renversées. On définira deux nouveaux motifs pour simuler un coureur qui saute de temps en temps par dessous une haie, et qui court le reste du temps. Ces deux motifs prendront la place des crochets [et].

La frappe d'une touche suffit pour faire sauter le coureur, à condition de s'y prendre au bon moment, juste avant la haie. Si on tape trop tôt la tentative de saut est un échec, et le coureur n'aura pas le temps de récupérer avant la haie : il va la renverser !

```
10 CLS:PAPER 0:INK 7
20 REPEAT
30 INPUT"VITESSE (1 A 20)";V
40 UNTIL V>0 AND V<11
50 CLS:LORES 0
60 D=11-V
70 NH=0:N=0:T=0
80 H$="I
85 :
90 PLOT 1,11,"0  HAIE RENVERSEE"
100 PLOT 23,11,"VITESSE"+STR$(V)
110 GOSUB 370
120 PLOT0,2,"DEPART"
130 PLOT24,10,"ARRIVEE"
140 FOR L=3 TO 9 STEP 2
150 PLOT 12-L,L,H$+H$+H$
160 NEXT L
170 WAIT 200
175 :
```

REMARQUES

20 à 40 — introduction de la vitesse du coureur avec vérification

60 — D : délai réglant la vitesse

70 — NH : nombre de haies, N : nombre de foulées depuis la dernière haie, T : indicateur de tentative

80 — une haie

90, 100 — tableau d'affichage

110 — utilisation du sous-programme de définition des deux motifs "coureur"

120 à 160 — dessin du parcours

```
180 SHOOT
190 FOR L=3 TO 9 STEP 2
200 FOR C=0 TO 31 STEP 2
210 PLOT C,L,"]"
220 WAIT D
230 PLOT C,L," "
240 K=0:N=N+2
250 T$=KEY$
260 IF T=0 AND N=10 AND T$<>"" THEN L=
L-1:K=1
270 IF (N<10) AND (T$<>"") THEN T=1
280 WAIT D
290 PLOT C+1,L,"["
300 IF N=10 THEN GOSUB 460
310 IF (K=0) AND (N<>0) THEN WAIT D:PL
OT C+1,L," "
320 IF K=1 THEN WAIT D:PLOT C+1,L," ":
L=L+1
330 NEXT C
340 NEXT L
350 END

360 REM-----------------------
370 DATA 8,28,8,29,42,24,20,34
380 DATA 8,28,8,29,42,40,20,2
390 FOR CO=91 TO 93 STEP 2
400 FOR I=0 TO 7
410 READ A
420 POKE 46080+CO*8+I,A
430 NEXT I
440 NEXT CO
450 RETURN

455 REM-----------------------
460 N=0:T=0
470 IF K=1 THEN RETURN
480 PING
490 NH=NH+1
500 PLOT C+1,L,"-"
510 PLOT 0,11,STR$(NH)
520 IF NH=2 THEN PLOT 8,11,"S RENVERSE
ES"
530 RETURN
```

REMARQUES

180 — le signal du départ

190 à 340 — boucle de parcours des 4 lignes droites (positions sur l'écran : 3, 5, 7, 9)

200 à 330 — boucle de parcours d'une ligne droite

210 — le coureur est dans une position

220, 230 — pause réglant la vitesse : effacement du coureur

240 — K indique que le coureur vient de sauter : 0 = non, 1 = oui

250 — frappe de touche ?

260 — le coureur atteint une haie et on n'a pas appuyé sur une touche trop tôt : si on appuie maintenant, le saut va avoir lieu

270 — on désarme T que l'on réamorce aussitôt en cas de frappe de touche avant d'avoir atteint une haie

280 — pause

290 — autre position

300, 310 — si le coureur ne vient pas de sauter et ne se trouve pas sur une haie, on l'efface

320 — si le courant vient de sauter, il faut l'effacer et le faire redescendre !

370 à 450 — sous-programme de définition des 2 motifs "coureur"

370, 380 — description des motifs

390 à 440 — boucles de lecture de cette description, touches [et]

460 à 530 — sous-programme de renversement d'une haie

460 — remise à zéro du nombre de foulées depuis la dernière haie et de l'indicateur de tentative

470 — si le coureur passe au-dessus, on ne touche pas la haie

480 — la cloche

490 — on ajoute 1 au nombre de haies renversées

500 — renversement

510 — affichage du score

520 — à partir de la deuxième haie renversée, il faut le pluriel

Faites un essai avec la vitesse 1 pour commencer. Ce n'est pas si simple :

Avant le départ.

```
DEPART
        I          I          I

      I          I          I

    I          I          I

  I          I          I
                         ARRIVEE
0  HAIE RENVERSEE       VITESSE1
```

Le coureur s'élance.

```
DEPART
     ☆  I          I          I

      I          I          I

    I          I          I

  I          I          I
                         ARRIVEE
0  HAIE RENVERSEE       VITESSE1
```

La première haie est franchie.

```
DEPART   ☆
         I          I          I

      I          I          I

    I          I          I

  I          I          I
                         ARRIVEE
0  HAIE RENVERSEE       VITESSE1
```

Le coureur s'est déconcentré et a renversé 4 haies ; il approche de la sixième...

```
DEPART
         I          -          -

      -          -     ☆    I

    I          I          I

  I          I          I
                         ARRIVEE
4  HAIES RENVERSEES     VITESSE1
```

C'est l'arrivée : il n'y a pas eu d'autres chutes.

```
DEPART
          I          -          -

        -          -          I

      I          I          I

    I          I          I
                          ARRIVEE
4   HAIES RENVERSEES    VITESSE1
```

35

L'ARDOISE MAGIQUE

L'Oric a un graphisme suffisamment fin pour inciter à l'utiliser comme ardoise. Au déplacement classique dans les quatre sens, nous ajouterons le tracé de segment et la possibilité de lever le ''crayon''.

```
10 HIRES
20 A=1:X=120:Y=100
30 PRINT"ARDOISE MAGIQUE"
40 :
50 REPEAT
60 :
70 REPEAT
80 CURSET X,Y,2:WAIT 1:CURSET X,Y,2
90 T$=KEY$
100 UNTIL T$<>""
110 :
120 C=ASC(T$)
130 IF T$="E" THEN A=0
140 IF T$="A" THEN A=1
150 X=X+(C=8)-(C=9)
160 Y=Y-(C=10)+(C=11)
170 IF X<0 THEN X=0
180 IF X>239 THEN X=239
190 IF Y<0 THEN Y=0
200 IF Y>199 THEN Y=199
210 CURSET X,Y,A
220 UNTIL C=3
```

On peut aussi remplacer ces 2 lignes par 90 GET T$

REMARQUES

20 — crayon baissé, position horizontale et verticale du crayon

50 à 220 — répétition jusqu'à ce qu'on tape CTRL-C

70 à 100 — attente d'une frappe de touche

80 — la pointe du crayon clignote

90 — T$ contient la dernière frappe de touche

100 — pas de touche frappée : on en attend une

120 — C$: code de la touche frappée

130 — on efface

140 — on trace

150, 160 — selon la touche enfoncée, on prépare le déplacement du crayon en

170 à 200 — rejet des positions horizontales et verticales situées hors de l'écran

210 — marquage ou effaçage du point

220 — à suivre !

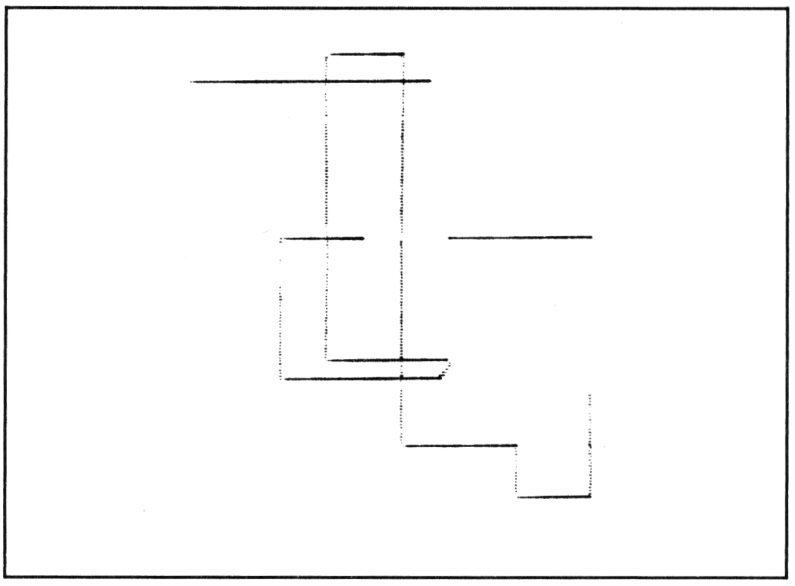
Mode d'emploi :

Après avoir tapé RUN, un point blanc apparaît au centre de l'écran ; le crayon est prêt à tracer si l'on appuie sur l'une des flèches.
Pour lever le crayon, taper E, et pour l'abaisser, taper A.

Le programme n'a pas de fin naturelle ; il suffit de taper CTRL C pour en avoir raison. Si vous êtes satisfait de votre œuvre, n'oubliez pas que vous pouvez la conserver sur cassette et lui donner un nom grâce à :

CSAVE ''IMAGE'', A40960, E49120 pour enregistrer à vitesse élevée,

ou CSAVE ''IMAGE'', A40960, E49120, S pour la petite vitesse.



Comment améliorer cette ardoise

Avec la version que nous venons de donner, il est difficile de tracer des obliques, et lorsqu'on se déplace sans tracer, c'est toute une gymnastique de ne pas effacer ce qui existe déjà.

Nous allons offrir quatre possibilités supplémentaires :

— garder les points allumés
— se déplacer rapidement
— mettre en mémoire la position d'un point
— relier ce point au point actuel.

Vous pouvez même changer de couleur en cours de route. Devant toutes ces possibilités, vous saurez où vous en êtes grâce à un tableau de bord.

```
10 HIRES:INK 7
20 A=1:V=1:X=120:Y=100:XM=120:YM=100
30 PRINT"ARDOISE MAGIQUE":PRINT"POINTS A
LLUMES, VITESSE LENTE"
40 :
50 REPEAT
60 :
70 REPEAT
80 CURSET X,Y,2:WAIT 1:CURSET X,Y,2
90 T$=KEY$
100 UNTIL T$<>""
110 :
120 C=ASC(T$)
130 :
140 IF T$="E" THEN A=0
150 IF T$="A" THEN A=1
160 IF T$="G" THEN A=3
170 IF T$="M" THEN XM=X:YM=Y
180 IF T$="R" THEN V=5
```

REMARQUES

20, 30 — au départ !

50 à 420 — répétition de tout ce qui suit jusqu'à ce qu'on frappe CTRL C

70 à 100 — attente d'une frappe de touche et clignotement du point

120 — C : le code de la touche

140 à 200 — étude du choix

140, 150 — points éteints ou allumés

160 — garder les points existants, mais ne pas en allumer d'autres

170 — mise en mémoire de la position actuelle

180, 190 — vitesse rapide ou lente

```
190 IF T$="L" THEN V=1
200 IF T$="T" THEN DRAW XM-X,YM-Y,A
210 :
220 IF C>=48 AND C<=55 THEN INK C-48
230 :
240 X=X+V*((C=8)-(C=9))
250 Y=Y-V*((C=10)-(C=11))
260 IF X<12 THEN X=12
270 IF X>239 THEN X=239
280 IF Y<0 THEN Y=0
290 IF Y>199 THEN Y=199
300 :
310 CURSET X,Y,A
320 :
330 CLS
340 PRINT"POINTS  VITESSE  MEMOIRE  POSI
TION"
350 IF A=0 THEN PRINT"ETEINTS ";
360 IF A=1 THEN PRINT"ALLUMES ";
370 IF A=3 THEN PRINT"GARDES  ";
380 IF V=1 THEN PRINT"LENTE  ";
390 IF V=5 THEN PRINT"RAPIDE ";
400 PRINT SPC(3);XM;YM;SPC(3);X;Y
410 :
420 UNTIL C=3
```

REMARQUES

200 — relier le point actuel au point mémorisé (pour allumer ou effacer un trait)

220 — choix de la couleur

240, 250 — positions horizontale et verticale

260 à 290 — butées gauche, droite, haute et basse

310 — nouveau positionnement

330 à 400 — tableau de bord mis à jour, sur deux lignes en bas de l'écran

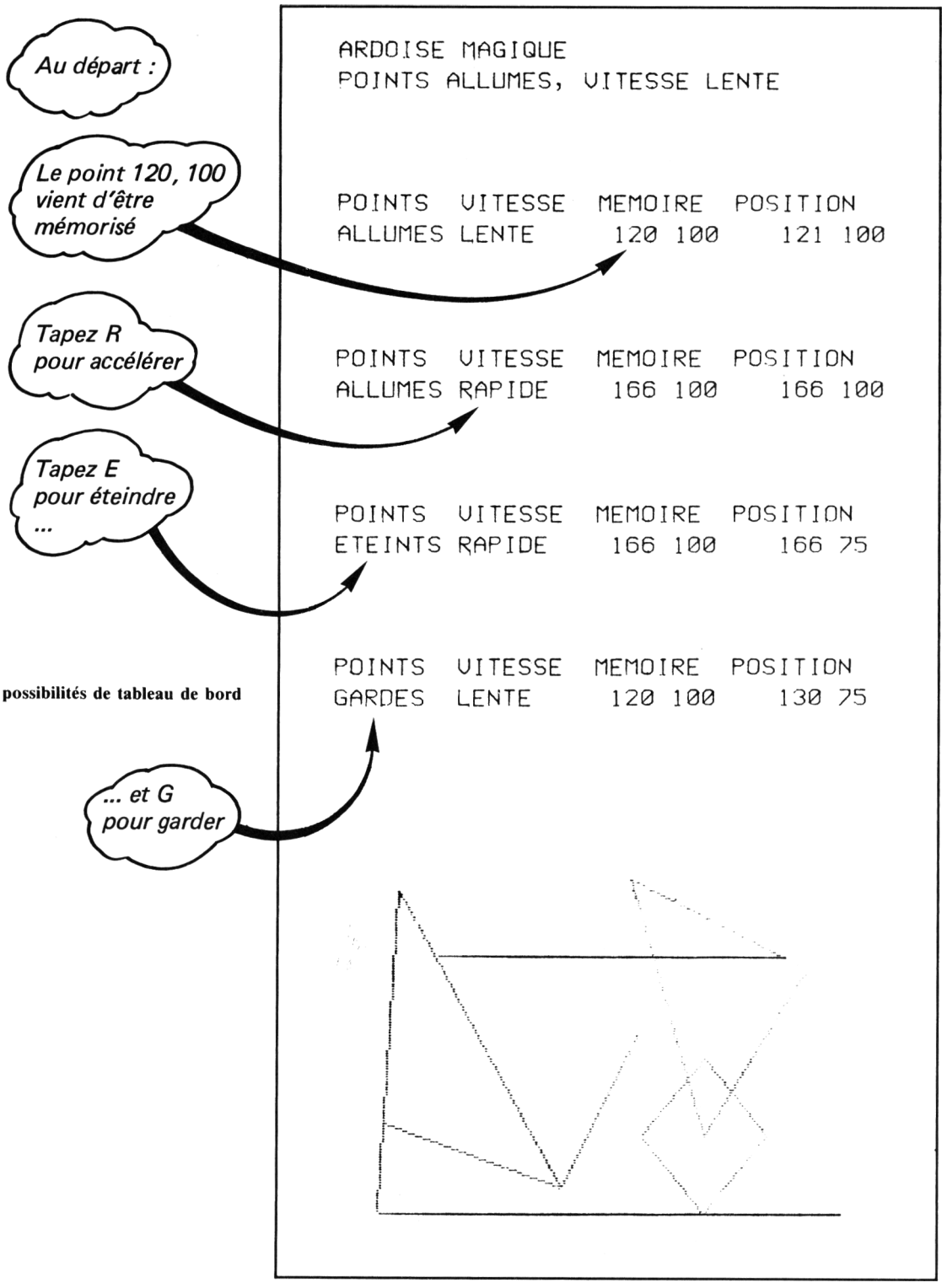
Mode d'emploi :

Pour se déplacer, c'est toujours les quatre flèches, mais si vous avez choisi la vitesse rapide, le point sautera cinq fois plus vite qu'avant. Pour tracer un trait dans n'importe quelle direction placez-vous à une des extrémités, mémorisez votre position en tapant M, déplacez-vous vers l'autre extrémité puis demandez le tracé en tapant T. Si c'est une oblique que vous voulez, déplacez-vous sans rien tracer en tapant G avant de partir, mais il faudra penser à rallumer (A) avant de tracer.

Pour effacer un point tapez E et pour effacer un trait tapez également E avant T.

Si vous voulez changer de couleur, il suffit de taper un des chiffres 0, 1,... 7 selon le code convenu pour l'Oric.

Les positions 0 à 11 posent des problèmes quand on fixe la couleur. Nous ne les utiliserons pas.



Quelques possibilités de tableau de bord

36

ORGUE À MÉMOIRE

Il est facile de produire un son en utilisant l'instruction MUSIC : on obtient chaque fois un son pur avec une graduation par demi-ton. Il est alors tentant de transformer le clavier en pupitre d'orgue, en toute modestie !

Nous allons le faire sur deux octaves, à un ton près :

Q	W	E	R	T	Y	U	I	O	P
mi	fa	fa #	sol	la^b	la	si^b	si	do	do #

A	S	D	F	G	H	J	K	L
sol	la^b	la	si^b	si	do	do #	ré	mi^b

Z	X	C	V	B	N	M
do	do #	ré	mi^b	mi	fa	fa #

espace

silence

Au fur et à mesure que vous pianoterez, les notes apparaîtront sur l'écran à leur hauteur respective, la plus basse (touche Z) en bas de l'écran.

```
10 CLS:PRINT"UN CLAVIER MUSICAL"
20 INPUT"VOLUME";V
30 INPUT"OCTAVE BASSE";OB
40 DATA Z,X,C,V,B,N,M,A,S,D,F,G,H,J,K,
L,Q,W,E,R,T,Y,U,I,O,P
50 DIM NO(90),OC(90)
60 N=1:O=OB
70 FOR L=1 TO 26
80 READ C$:C=ASC(C$)
90 NO(C)=N:OC(C)=O
100 N=N+1
110 IF N=13 THEN O=O+1:N=1
120 NEXT L
125 :
```

REMARQUES

20, 30 — introduction du volume et de l'octave basse

40 — les lettres selon leur place sur le clavier de bas en haut et de gauche à droite

50 — réservation pour les fichiers des notes et des octaves

60 — note et octave basses

70 à 120 — boucle de correspondance entre les codes des lettres et les sons

80 — C : code d'une touche, selon la disposition fournie par l'instruction DATA

90 — mémorisation de la note et de l'octave correspondante

110 — passage à l'octave supérieure

```
130 F=1:CLS
140 REPEAT
150 GET T$
160 IF F=39 THEN F=1:CLS
170 IF T$>="A" AND T$<="Z" THEN GOSUB
300
180 F=F+1
190 UNTIL T$=CHR$(3)
200 END
300 REM----------------------
310 T=ASC(T$)
320 MUSIC 1,OC(T),NO(T),V
330 H=27-(OC(T)-OB)*12-NO(T)
340 PLOT F,H,"!"
350 RETURN
```

REMARQUES

140 à 190 — répéter ce qui suit jusqu'à ce qu'on frappe CTRL C

150 — attente d'une frappe de touche

160 — changement de "page écran"

170 — si on a frappé une lettre : musique

310 à 350 — sous-programme de production d'un son

310 — code de la touche

320 — le son

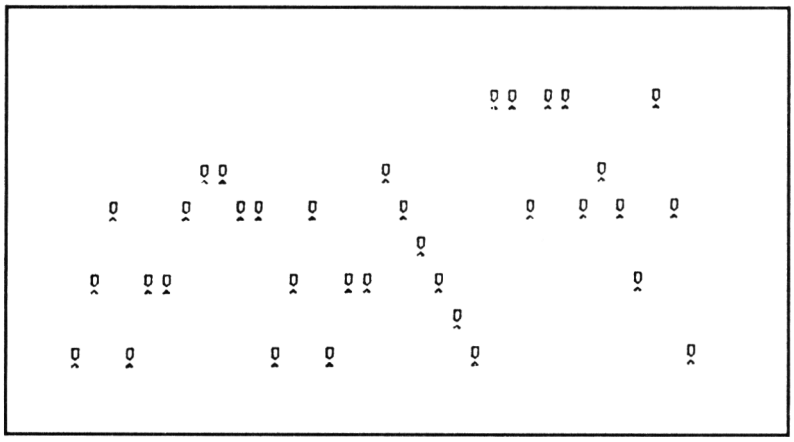
330 — hauteur de ce son, sur l'écran (en bas : grave, en haut : aigu)

340 — affichage d'un point d'exclamation au bon endroit sur l'écran

Choisissez le volume et l'octave basse :

```
UN CLAVIER MUSICAL
VOLUME? 4
OCTAVE BASSE? 3
```

et composez ou reproduisez un air connu !



VERSION AVEC MEMOIRE

Pour accroître l'intérêt du jeu tout en gardant un programme simple, nous allons mémoriser les hauteurs de chaque note, de façon à ré-écouter un air que l'on aura composé au clavier. On pourra fixer la durée.

```
 10 CLS:PRINT"CLAVIER MUSICAL A MEMOIRE
"
 20 INPUT"VOLUME";V
```

```
30 INPUT"OCTAVE BASSE";OB
40 DATA Z,X,C,V,B,N,M,A,S,D,F,G,H,J,K,
L,Q,W,E,R,T,Y,U,I,O,P
50 DIM NO(90),OC(90)
55 DIM T$(5,39)
60 N=1:O=OB
70 FOR L=1 TO 26
80 READ C$:C=ASC(C$)
90 NO(C)=N:OC(C)=O
100 N=N+1
110 IF N=13 THEN O=O+1:N=1
120 NEXT L
125 :
130 M=1:F=0:CLS
135 PLAY 1,0,0,0
140 REPEAT
150 GET TT$
160 IF TT$="0" THEN GET TT$:PLOT F,H,"
 " ELSE F=F+1
170 IF F=39 THEN M=M+1:F=1:CLS
175 T$(M,F)=TT$
180 IF T$(M,F)>="A" AND T$(M,F)<="Z" T
HEN GOSUB 300
190 UNTIL TT$=CHR$(3)
200 :
210 MM=M:FF=F
220 M=1:F=1:CLS
230 REPEAT
240 IF F=39 THEN M=M+1:F=1:CLS
245 IF T$(M,F)=" " THEN MUSIC 1,0,1,0
250 IF T$(M,F)>="A" AND T$(M,F)<="Z" T
HEN GOSUB 300
255 WAIT 20
260 F=F+1
270 UNTIL M=MM AND F=FF
275 :
280 PLAY 0,0,0,0
290 END
300 REM----------------------
310 T=ASC(T$(M,F))
320 MUSIC 1,OC(T),NO(T),V
330 H=27-(OC(T)-OB)*12-NO(T)
340 PLOT F,H,"▯"
350 RETURN
```

REMARQUE

55 — réservation pour les frappes de touches à enregistrer

REMARQUES

130 — premier morceau

135 — silence

150 — attente d'une frappe de touche

160 — effacement d'une erreur si on raye zéro, sinon passage au prochain son

170 — passage au morceau suivant

175 — mémorisation de la touche

180 — production du son si c'est une lettre qui a été frappée

210 à 270 — écoute de l'air enregistré

210 — MM : le nombre de morceaux, FF : le nombre de notes du dernier morceau

220 — premier morceau

230 à 270 — répéter ce qui suit jusqu'au dernier son enregistré

240 — passage au morceau suivant

245 — un silence

250 — un son

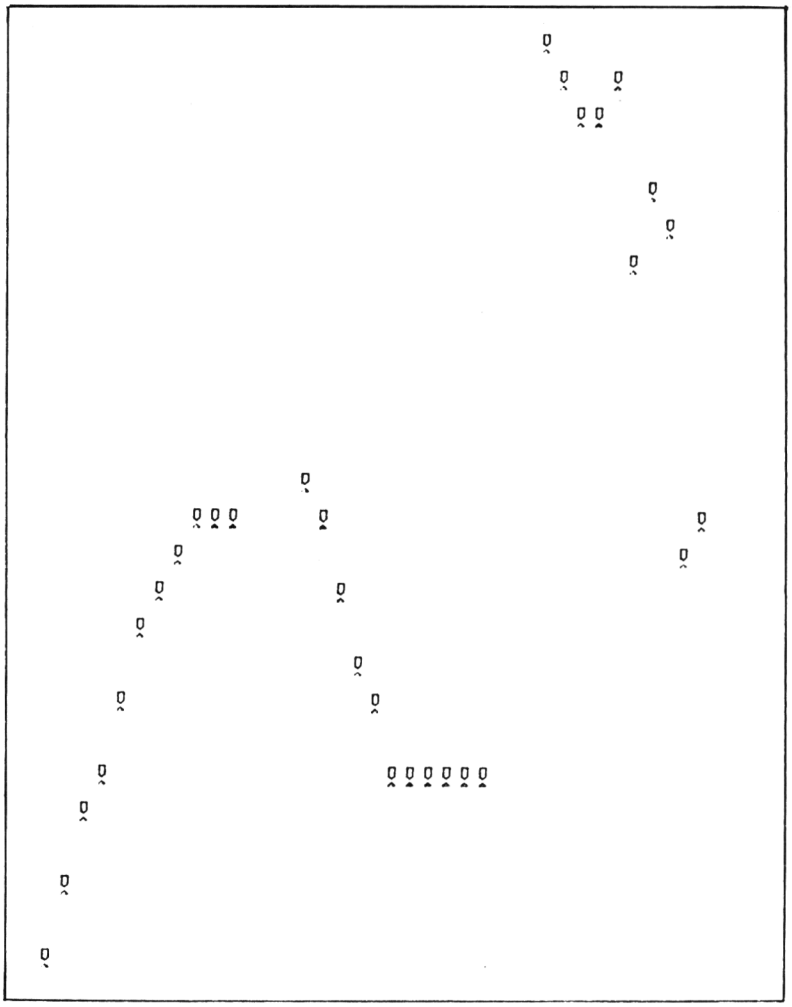
255 — durée de ce son ou de ce silence : 2/10 de seconde

280 — arrêt de la dernière note

Après le choix du volume et de l'octave basse.

```
CLAVIER MUSICAL A MEMOIRE
VOLUME? 1
OCTAVE BASSE? 2
```

Vous pouvez pianoter, avec droit à l'erreur (en tapant zéro vous gommez la fausse note). Mais il n'y a pas de sécurité sur la longueur de l'air enregistré : ne dépassez pas 5 morceaux.



La première écoute est gratuite, pour les suivantes il faudra faire GOTO 220.

Et si vous le souhaitez vous pouvez même changer le volume et décaler l'air entier vers les graves ou les aigüs d'une ou plusieurs octaves, sans retaper les notes :

```
V=8
OB=5
GOTO 220
```

Les programmes que nous vous avons donnés avaient comme but de vous mettre en route. Vous devez pouvoir maintenant écrire les vôtres et tirer profit de ceux que vous trouverez dans le commerce en les comprenant et en les modifiant.

LE BASIC DE L'ORIC : LES MOTS RÉSERVÉS

Voici la liste des instructions Basic utilisées par l'Oric. Nous avons marqué en gras celles qui ont été étudiées dans ce livre, car on ne les a pas toutes passées en revue. En effet, on a abandonné certaines fonctions mathématiques, *ainsi que des instructions spécifiques au* langage « machine ». *Pour tout cela, vous pourrez maintenant vous reporter au manuel d'emploi de l'Oric qui est à la fois presque complet (lui aussi !) mais malheureusement généralement hermétique. Mais vous savez maintenant utiliser l'essentiel des commandes, et vous ne serez plus guère désorienté.*
Si vous voulez retrouver la page à laquelle chaque instruction étudiée est développée, reportez-vous à la liste des « Mots clés » qui la précise.
Toutes ces commandes Basic sont des « mots réservés ». Vous n'avez pas le droit de les employer comme noms de variables, ni même les inclure dans des noms de variables.

ABS
AND
ASC

ATN
CALL
CHAR

CHR$
CIRCLE
CLEAR

CLS
CLOAD
CONT

COS
CURMOV
CURSET

CSAVE
DATA
DEEK

DEF FN
DEF USR
DIM

DOKE
DRAW
EDIT
ELSE
END
EXP

EXPLODE
FALSE

FILL
FN
FOR... TO

FOR... TO... STEP
FRE
GET

GOSUB
GOTO
GRAB

HEX$
HIMEM
HIRES

IF... THEN
IF... THEN... ELSE
INK

INPUT
INT
KEY$

LEFT$
LEN
LET

LIST
LLIST
LN

LOG
LORES
LPRINT

MID$
MUSIC
NEW

NEXT
NOT
ON... GOSUB

ON... GOTO
OR
PAPER

PATTERN
PEEK
PI

PING
PLAY
PLOT

POINT
POKE
POP

PRINT
PULL
READ

RELEASE
REM
REPEAT

RESTORE
RETURN
RIGHT$

RND
RUN
SCRN

SGN
SHOOT
SIN

SOUND
SPC
SQR

STEP
STOP
STR$

TAB
TAN
TEXT

TROFF
TRON
TRUE

UNTIL
USR
VAL

WAIT
ZAP

LE CODE ASCII

ASCII signifie « American Code for Information Interchange », soit code américain pour l'échange d'informations. *Chaque frappe, simple ou combinée, trouve sa traduction dans ce code. Voici ce code, dans sa normalisation officielle, c'est-à-dire sans les caractères graphiques personnels à l'Oric.*

CODIFICATION A.S C.I.I.

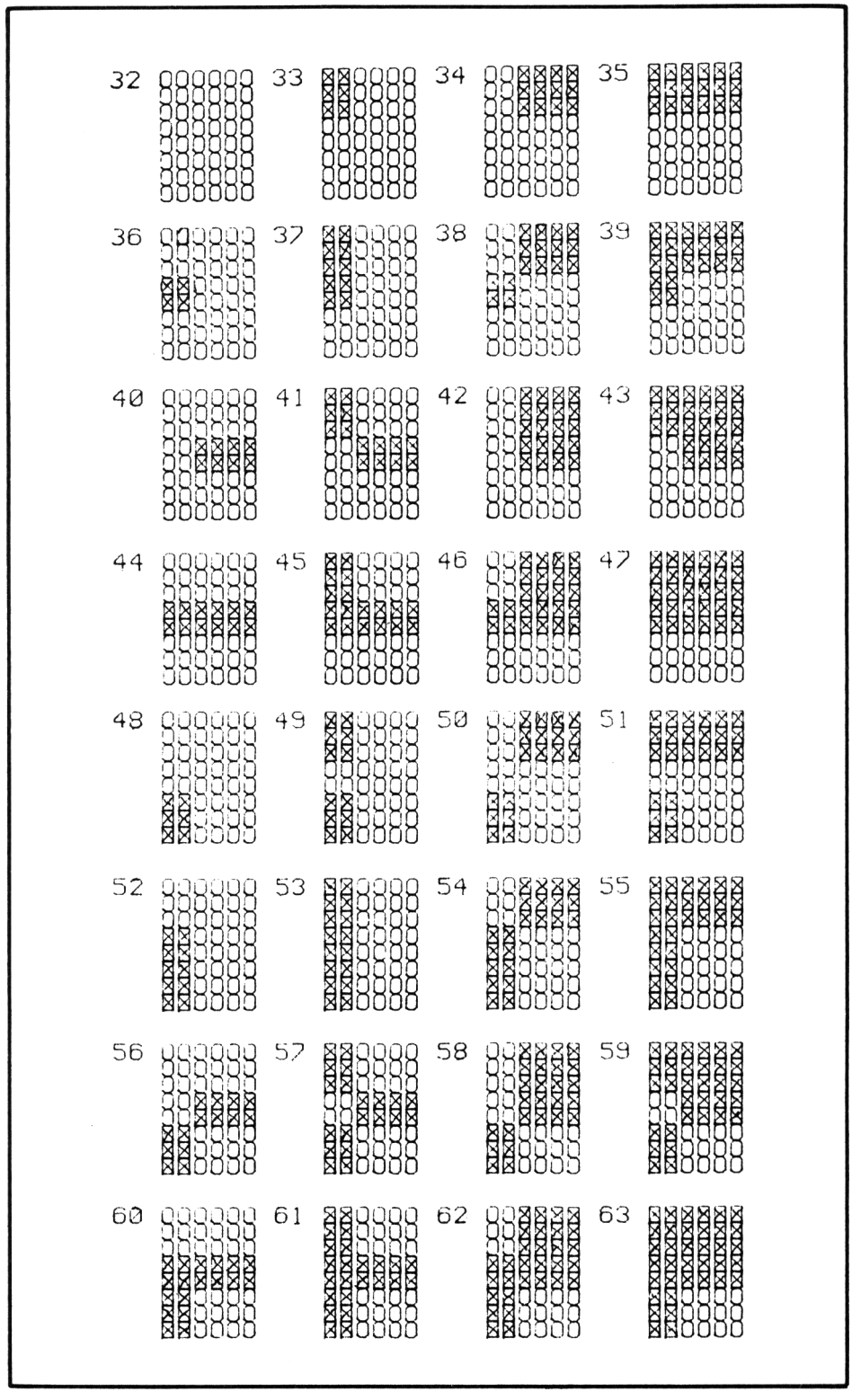
32	Espace	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_	127	DEL

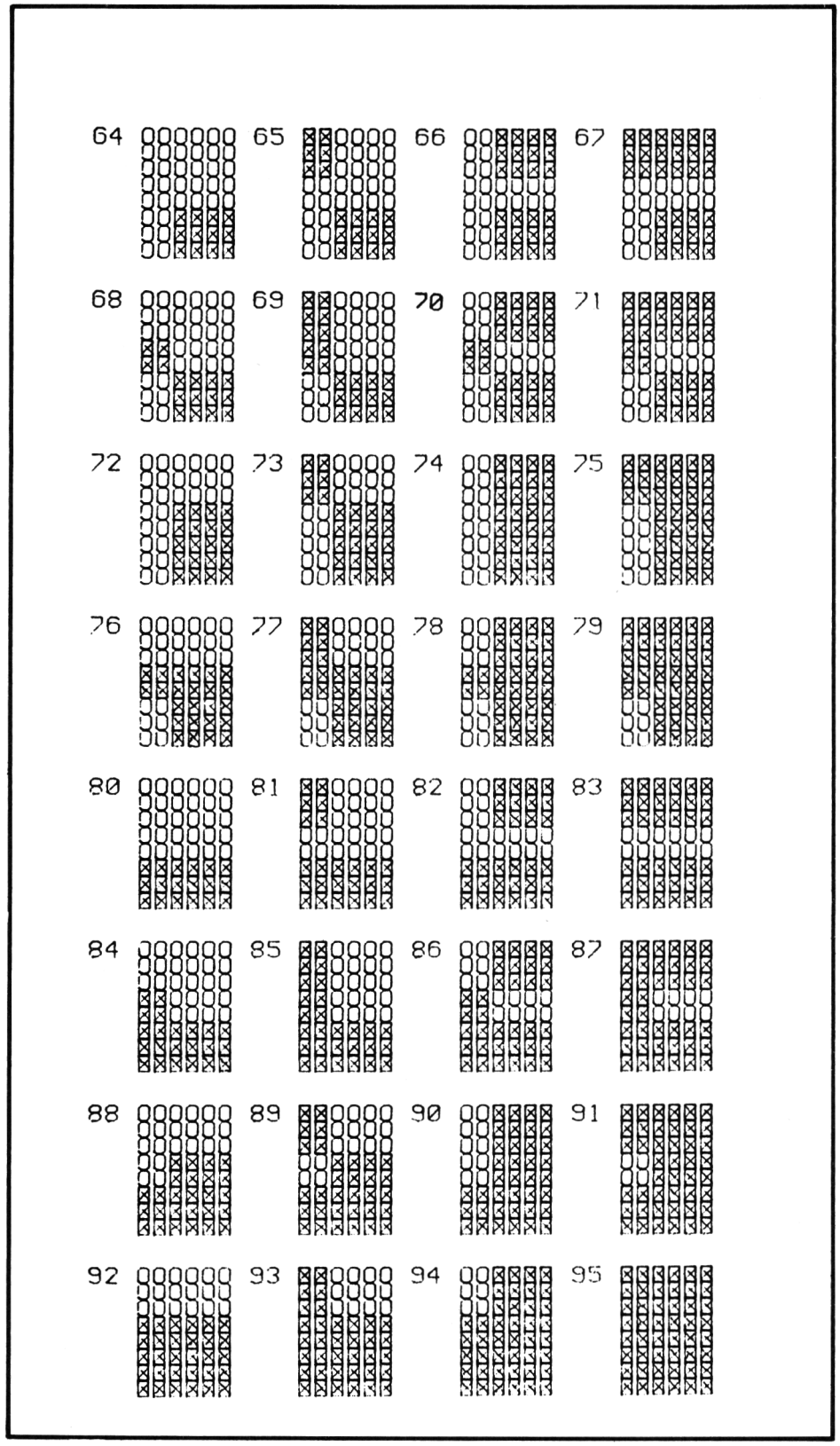
LES CARACTÈRES GRAPHIQUES DE L'ORIC

Lorsqu'on choisit le jeu de caractères numéro 1, certains codes (32 à 95) fournissent des caractères utiles pour créer des images graphiques basse résolution. Voici ces 64 caractères, agrandis afin de situer les zones d'encre par rapport au fond.

Un 0 représente un point éteint (couleur du fond)

Un ⊠ représente un point allumé (couleur de l'encre)




64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95

MOTS CLÉS

En marge du texte, nous avons encadré des « mots clés » importants. La liste vous en a été fournie, chapitre par chapitre, au début de chacun d'eux. Les voici maintenant par ordre alphabétique, avec renvoi à la ou les pages où ils sont définis.

A
Accord musical, 222
Adjonction d'une ligne, 46
Affectation, 40
Alimentation, 8
AND, 145
Annulation d'une ligne, 34
Attributs, 147
AZERTY, 10

B
Barres de couleur, 105
Basic, 12, 13
Bataille galactique, 85
Boucles, 73, 80, 82
Branchement de l'Oric – 1, 7
Branchement conditionnel, 145
Branchement inconditionnel, 54, 145
Byte, 14

C
Canaux des sons, 217
CAPS, 16
Caractère, 14, 39
Caractères double hauteur, 148
Caractères graphiques, 115
Carré en LORES Ø, 110
Cellules mémoires, 14
Cercles à géométrie variable, 139
Chaînes de caractères, 39, 42
Chaîne vide, 42
CHAR, 140
Chargement d'un programme, 21
Chiffre Ø et lettre O, 13
CHR$, 56
CHR$(8), 117
CHR$(9), 117
CIRCLE, 137
Clignotement, 148
CLOAD, 21
CLS, 54
Code ASCII, 56
Concaténation de chaînes, 43
Connexion d'un magnétophone, 19
Constante, 40
Constante chaîne, 40
Constante numérique, 40
CONT, 55, 116
Couleurs, 78
Couleurs avec PLOT, 103
CSAVE, 25
CSAVE AUTO, 51
CTRL, 16
CTRL/A, 58
CTRL/C, 53, 116
CTRL/D, 148
CTRL/F, 43
CTRL/G, 82
CTRL/H, 38
CTRL/I, 38
CTRL/J, 38
CTRL/K, 38
CTRL/L, 38
CTRL/M, 38
CTRL/N, 44
CTRL/Q, 111
CTRL/T, 16
CTRL/X, 34
CURMOV, 135
CURSET, 135
Curseur, 11

D
DATA, 121, 122
Décomptage avec FOR... TO., 78
DEEK, 151
Défilé des couleurs, 123
DEL, 11
Démarrage à chaud, 24
Démarrage à froid, 24
Démarrage automatique, 29
DIM, 127
Division, 36
DOKE, 151
Double hauteur, 207
Double-point, 76
DRAW, 137
Duplication d'instructions, 109

E
Ecran haute résolution, 135
Edit, 59
Effacement arrière, 11
Elévation à puissance, 37
Effacement de la mémoire, 49
Effacement de l'écran, 38

Effacement d'une ligne, 44
Effacement programmé de l'écran, 54
ELSE, 145
Emboîtage de boucles, 80
END, 112
Entiers, 132
Enveloppe des sons, 218
Error, 11
Etoiles, 111
EXPLODE, 83

F
FILL, 141
Format de l'écran, 16
FOR... TO, 73

G
GET, 226
GOSUB, 113
GOTO, 52
Graphique basse résolution, 105
Grille graphique basse résolution, 102
Guillemets, 31

H
Haute résolution, 133, 135
HIRES, 106, 133

I
IF... THEN, 144
IF... THEN... ELSE, 200
INK, 78
INPUT, 50
Instructions multiples, 76
INT, 132

K
KEY$, 146

L
Langages, 12
Lecture d'une touche, 146
LEFT$, 67
LEN, 67
Lignes de couleurs aléatoires, 142
Lignes doublées, 148
Ligne logique, 15
Ligne physique, 24
LIST, 47
Listage partiel, 57
LORES, 106
LORES Ø, 106
LORES 1, 114

M
Majuscules, 16
Mémoires centrales, 13
Mémoires externes, 14
Mémoires internes, 13
Mémoires mortes, 14
Mémoires, 14, 49
Mémoires périphériques, 14
Mémoires vives, 14
Messages d'erreur, 49
Microprocesseur, 13
Microsoft, 13
MID$, 68
Minuscules, 16
Mise en service, 9
Mode direct, 31, 35
Mode programmé, 47
Mode texte, 101
Modes graphiques, 101
Modification d'une ligne, 58
Mots réservés, 42
Mot transformé, 129
Mouvement, 116
Multiplication, 36
MUSIC, 213

N
NEXT, 73
NEW, 49
Nombres aléatoires, 83
Noms de variables, 71
NOT, 145
Notation scientifique, 94
Notes musicales, 214
Numérotation des lignes, 45

O
Octet, 14
Opérateurs relationnels, 145
OR, 145
Organigramme, 52
Organigramme de FOR... TO, 74

P
PAPER, 78
PATTERN, 139
Pas dans FOR... TO, 76
PEEK, 152
Période avec SOUND, 224
PING, 82
PLAY, 213, 217
PLOT, 102
POINT, 140
Points d'affichage, 135
Point décimal, 36
Point d'interrogation pour PRINT, 34
Point-virgule, 35
POKE, 152
Pour sauter une ligne, 95
PRINT, 32
Prise Péritel, 8
Programme de conversion, 50
Programme principal, 112

Q
QWERTY, 10

R
Racine carrée, 37
RAM, 21
READ, 14
REDO FROM START, 53
Règles avec les boucles, 82
Ré-initialisation, 23
REM, 79
Remarques, 78
Remplacement d'une ligne, 48

REPEAT UNTIL, 144
Répétition des touches, 12
RESET, 24
RETURN, 17, 32, 113, 130
RIGHT$, 67
RND, 83
RND(Ø), 86
RND(−N), 86
Rôle des guillemets, 35
ROM, 14
RUN, 46
RUN numéro de ligne, 111

S
Sauter une ligne, 55
Sauvegarde d'un programme, 25
SCRN, 129
SGN, 124
SHIFT, 12
SHOOT, 83
Sonorisation des touches, 43
Sons, 82
SOUND, 213, 224
Sous-programme, 112
SQR, 37
STEP, 76
STR$, 69
STOP, 112
Suppression d'une ligne, 48
Suppression du curseur, 111
Symbole d'affectation (=), 40
Symbole de concaténation, 43
Symbole dollar, 39
Syntaxe, 35
SYNTAX ERROR, 12

T
Temporisation, 59
TEXT, 101, 134, 135
Traits en LORES Ø, 108

V
VAL, 69
Variable, 40
Variable chaîne, 40
Variable numérique, 40
Virgule dans un PRINT, 37
Virgule décimale, 36
Vitesse lente, 22
Volatilité des mémoires, 14

W
WAIT, 59

Z
ZAP, 83
Zéro et lettre O, 36

TABLE DES MATIÈRES

Avant-propos 5
Chapitre I — **Pour faire connaissance avec l'Oric** 7
1. L'Oric – 1 et ses branchements 7
2. Mise en service 9
3. Qu'est-ce que le Basic ? 12
4. Microprocesseurs et mémoires 13
5. Clavier et écran 15

Chapitre II — **Magnétophone et cassettes de programmes** 19
1. Le magnétophone 19
2. Chargement d'un programme 20
3. Sauvegarde d'un programme 25
4. Les règles du chargement 27

Chapitre III — **L'ordinateur en « mode direct »** 31
1. L'instruction PRINT 31
2. Premiers calculs 35
3. Point-virgule et virgule 37
4. Variables et constantes 38
5. Chaînes de caractères 42

Chapitre IV — **Le mode programmé** 45
1. Numérotation des lignes 45
2. Les corrections 48
3. Effacement de la mémoire et de l'écran 49
4. Quelle est la vitesse du bateau ? 49
5. L'ordre d'entrée INPUT 50
6. Et si l'on recommençait ? 51
7. Effacement programmé de l'écran 54
8. Le code ASCII 56
9. L'édition 57

Programmes
1. Une addition qui n'en finit pas (CLS, INPUT, PRINT, GOTO, WAIT) 61
2. Un chronomètre digital (CLS, PRINT, WAIT, GOTO) 64
3. Caractères et codes : la norme ASCII (CLS, CHR$, ASC, PRINT, GOTO, WAIT) ... 65
4. L'initiale d'un mot (LEFT$, LEN) 67
5. Une tête à queue verbale (MID$) 68
6. Un compte rond (STR$, VAL) 69

Chapitre V — **Couleurs et sons programmés** 73
1. Boucles avec les ordres FOR... NEXT 73
2. Boucles FOR... TO avec un pas 76
3. Avec 8 couleurs 78
4. Des sons aussi 82
5. Nombres au hasard et bataille galactique 83

Programmes
7. Compte à rebours (FOR, NEXT, PING) 88
8. De zéro à 99 (FOR, NEXT) 89
9. Le hasard cet inconnu (FOR, RND, INT) 90
10. Pour réveiller, sons et couleurs ! (PING, ZAP, PAPER, IN) 95
11. Notation scientifique 1.23456689E + 10 97
12. Conversion binaire - décimale (FOR, INT) 99

Chapitre VI — **L'art graphique en basse résolution** 101
1. L'organisation de l'écran en « mode texte » 101
2. Graphique basse résolution, mode Ø 105
3. Ciel d'étoiles 109
4. Sous-programmes 112
5. Le mode LORES 1 114
6. Pour mêler les genres 117

Programmes
13. Arc-en-ciel (LORES, PLOT, READ, DATA) .. 121
14. C'est plus C'est moins (PLOT$, SGN) 124
15. Une mosaïque vivante (PLOT, LORES) 125
16. Histogramme (LORES, PLOT, DIM) 127
17. Un mot transformé (PLOT, SCRN) 129
18. Faire-part de naissance (PLOT, GO SUB, RETURN) 130

Chapitre VII — **Dessins en haute résolution** 133
1. Placement du curseur 133
2. Tracés 137
3. Couleurs en HIRES 141
4. Répétitions 143
5. Lecture d'une touche avec KEY$ 146
6. Les attributs 147

Programmes
19. Chronomètre digital précis (STR$, VAL) 151
20. Caractères accentués 154
21. Les caractères semi-graphiques à la loupe (HIRES, CURSET, CHAR, DRAW) .. 157
22. Création de motifs (HIRES, CURSET, CHAR, DRAW, DEF, FN) 159
23. Courbe digitalisée (HIRES, CURSET, DRAW) 163
24. Courbe paramétrée (HIRES, CURSET, DRAW, PATTERN, DEF, FN) 167
25. Diagramme circulaire (HIRES, CURSET, (CIRCLE, DRAW, CHAR) 173
26. FILL : du remplissage avec 64 motifs (HIRES, CURSET, CURMOV, FILL, CHAR) 175

Chapitre VIII — **Le moment musical** 177
1. Music, pour commencer 177
2. Pour jouer : PLAY 179
3. Des accords 181
4. Sons divers 182
5. Attendez une touche 183

Programmes
27. SOUND : découvrez les sons (SOUND, GET) 185
28. MUSIC : musique (MUSIC, GET) 186
29. SOUND MUSIC PLAY : pour mélanger les sons (SOUND, MUSIC, PLAY) 187
30. En rythme 191
31. Musique au hasard (MUSIC) 192

Chapitre IX — **Pour réviser cinq programmes utiles ou divertissants** 195

Programmes
32. Mise en forme : comment afficher proprement un nombre 195
33. Classement alphabétique 200
34. Course de haie 203
35. L'ardoise magique 207
36. Orgue à mémoire 211

Le Basic de l'Oric et les mots réservés 215

Le code ASCII 216

Les caractères graphiques de l'Oric 217

Mots-clés 219

S.E.C.F.

EDITIONS RADIO

Service lecteurs

(à retourner à S.E.C.F.-Éditions Radio, 9, rue Jacob, 75006 Paris)

Pour nous permettre de vous proposer des ouvrages toujours meilleurs, nous souhaiterions recevoir vos critiques, appréciations et suggestions sur le présent livre :

_____

_____

_____

_____

Quels sont les ouvrages (thème, sujet, niveau) que vous souhaiteriez voir publier par notre société ?

_____

_____

_____

_____

Nous vous remercions de votre confiance et de votre coopération.

S.E.C.F.-Éditions Radio

Je désire recevoir gratuitement et sans engagement (mettre une croix dans la case) :

☐ Votre catalogue général (Electronique professionnelle et grand public, Informatique, Hi-Fi, Vidéo)

☐ Votre catalogue spécial informatique.

Nom : _____ Prénom : _____

Adresse : _____

_____

Secteur d'activité et fonction : _____

_____

CENTRES D'INTÉRÊTS

☐ Electronique professionnelle

☐ Electronique de loisirs

☐ Vidéo

☐ Hifi, CB...

☐ Micro-informatique professionnelle

☐ Micro-informatique de loisirs

☐ Autres : .....

.....

Voir au dos "Correspondance Auteurs"

S.E.C.F.

EDITIONS RADIO

Service lecteurs

Correspondance auteurs

(à retourner à S.E.C.F.-Éditions Radio, 9, rue Jacob, 75006 Paris)

Pour toute demande d'éclaircissements techniques relatifs à ce livre, formulez ci-dessous vos questions, avec le maximum de précisions. Nous les transmettrons à l'auteur qui ne manquera pas de vous répondre directement :

Nom : _____ Prénom : _____

Adresse : _____

Le mensuel de la micro-informatique et de ses utilisations

en vente chez tous les marchands de journaux

S. E. C. F.

Editions Radio

9, RUE JACOB - 75006 PARIS

TÉL. 329.63.70

PRATIQUE de l'ORIC-1 et 36 PROGRAMMES

Micro-ordinateur familial par excellence, l'Oric-1 connaît un succès « explosif ». Ce livre est destiné à vous en présenter l'usage et vise un double objectif :

- Vous expliquer la programmation dans ce langage qu'on appelle le Basic et ce, à l'aide d'une infinité d'exemples pratiques.
- Vous proposer toute une série de programmes d'applications qui vous serviront à maîtriser parfaitement l'outil informatique.

Très progressif et écrit dans une langue simple, ce livre est aussi l'un des plus didactiques qui soient. Il ne demande au lecteur aucune connaissance préalable en informatique et est destiné à tous les publics.

9 782709 109321

ISBN 2 7091 0932 8
Code : 37

S. E. C. F.

ÉDITIONS RADIO

Prix : 100 F

PRATIQUE de l'ORIC-1 et 36 PROGRAMMES

H. LILEN - J. BÉNARD